Издательство «Знание»

ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА И ЕЁ ПРИМЕНЕНИЕ

Новое в жизни, науке, технике

Подписная научно-популярная серия

Издается ежемесячно с 1938 г.

Персональный компьютер «Электроника МК 85»

1992

2

Новое
в жизни,
науке,
технике

Подписная
научно-
популярная
серия

2/1992

Издается
ежемесячно
с 1988 г.

ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА И ЕЁ ПРИМЕНЕНИЕ

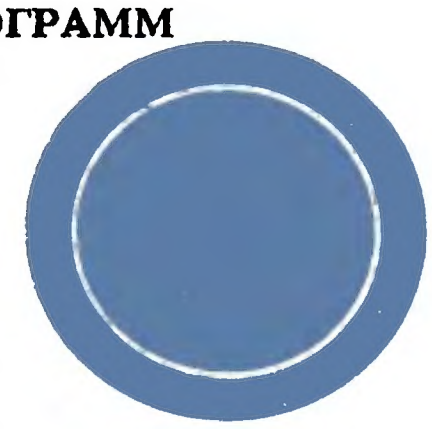
ПЕРСОНАЛЬНЫЙ КОМПЬЮТЕР «ЭЛЕКТРОНИКА МК 85»

В номере:

В.А.Хмелюк
САМЫЙ МАЛЕНЬКИЙ И
САМЫЙ ПЕРСОНАЛЬНЫЙ

В.А.Хмелюк
БИБЛИОТЕКА ПРОГРАММ

TUTTI FRUTTI



Москва
Издательство
«Знание»
1992

ББК 32.973
П 27

Авторы выпуска

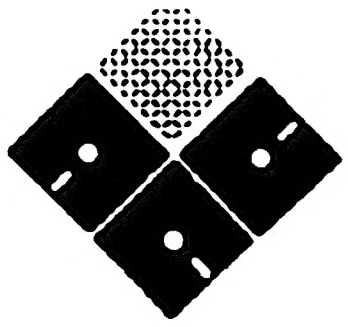
ХМЕЛЮК ВЛАДИМИР АРКАДЬЕВИЧ — кандидат технических наук, лауреат Государственной премии СССР, специалист в области космической радиосвязи.
ВУЛИС МИХАИЛ ЛАЗАРЕВИЧ, ВЯЗЕМСКИЙ КОНСТАНТИН ЯКОВЛЕВИЧ, ЗВЕРЕВ ДМИТРИЙ МАГОМЕДОВИЧ, ПОНОМАРЕНКО ВИКТОР ИВАНОВИЧ, СЕРБИЕНКО АНДРЕЙ ВЛАДИМИРОВИЧ

Редактор Б.М.Васильев

Scan AAW

ISBN 5-07-002247-4

© Издательство «Знание», 1992 г.



В нашем ЖУРНАЛЕ уже печатались программы для микрокомпьютера "Электроника МК-85", первого отечественного устройства класса pocketcomputer. По многочисленным просьбам читателей этот выпуск посвящен ему полностью. В нем сделана попытка удовлетворить как тех читателей, которые сетуют на отсутствие программ для МК-85 (публикуется более 39 программ), так и тех, кто просил дать подробное описание версии Бейсика для МК-85. Публикуются программы разных жанров, в том числе и такие, которые показывают, что возможности микрокомпьютера можно расширить по сравнению с указанными в "Руководстве по эксплуатации" (см.программы К.Вяземского и Д.Зверева). Все программы проверены на МК-85, но так как этот компьютер не имеет принтера, то готовить их для печати пришлось на IBM XT/AT. При ручном вводе программ в этот компьютер возникали ошибки, и нет стопроцентной гарантии того, что все они выловлены и устранены. О тех, которые найдут читатели, будет сообщено в следующих номерах (смотри новую рубрику "ОЧЕПАТКИ").В тех случаях, когда для обозначения одинаковых понятий в версии МК-85 и BASICA для IBM PC применяются разные символы, то в этом выпуске используются символы версии BASICA. В первую очередь это относится к символам отношений — не равно <>, равно или больше > =, равно или меньше < =. При отсутствии адекватного обозначения оно вводится описательно (π — PI, Ω — омега) или с помощью оператора CHR в сочетании с кодом соответствующего символа.

В.А.Хмелюк

САМЫЙ МАЛЕНЬКИЙ И САМЫЙ ПЕРСОНАЛЬНЫЙ

МАЛАЯ ЗАКВАСКА КВАСИТ ВСЮ ОПАРУ

Семейство самых маленьких вычислительных устройств обогатилось микрокомпьютером "Электроника МК-85" (в дальнейшем просто МК-85), выпускаемого в двух модификациях — МК-85 и МК-85м, отличающихся друг от друга только объемом ОЗУ программ и ценой. В редакцию поступает много писем как от владельцев, так и от потенциальных покупателей этих устройств. И если первых интересуют особенности эксплуатации и программирования, где достать литературу, программы и т.п., то вторых — какие задачи можно решать, что из себя представляет первый отечественный микрокомпьютер, где купить и сколько стоит. Конечно, можно попытаться ответить каждой категории отдельно, но ведь сегодняшние покупатели завтра станут владельцами и их будут волновать совсем другие вопросы. Эта статья — попытка ответить обеим категориям сразу. При этом считается, что читатель знаком с "Руководством по эксплуатации". Конечно, не имеющим руководства не все будет абсолютно ясно, но ведь и компьютера у них еще нет. В то же время для того чтобы изложение было связным, некоторые положения руководства пришлось повторить, и автор просит прощения у тех, у кого это вызовет досаду. Ну а ассам программирования, знакомым с языками высокого уровня, лучше всего не терять времени на эту статью.

МК-85 может работать в обычном калькуляторном режиме, и здесь его эксплуатация ничем не отличается от эксплуатации таких калькуляторов, как Б3-38 или МК-71. Работа с программой (ввод, редактирование, отладка и текстирование) в принципе не отличается от таких же операций при их проведении на персональном компьютере. Имеется разветвленная система диагностики, сообщающая пользователю об ошибках восьми типов, допущенных при вводе, неправильном использовании операторов и памяти, а также математических, синтаксических и иных ошибках.

Питается микрокомпьютер от четырех элементов СЦ 0,18 и от сети переменного тока (220 В, 50 Гц) через блок питания "Электроника Д2-10К". Потребляет в обычном

режиме не более 0,02 Вт. Существует режим быстродействия. Для перехода в него перед включением нажимают клавишу "+" и держат ее нажатой в момент включения питания. Быстродействие микрокомпьютера повышается в 4 — 6 раз, но и значительно повышается потребление. Не рекомендуется работать в режиме быстродействия при автономном питании, так как при этом быстро разряжаются элементы, а их комплект стоит около 13р [1]. Да еще и купить не всегда просто. А ведь элементы обеспечивают не только автономную работу, но и сохранность информации в энергонезависимой памяти, т.е. хранение при выключенном питании программ и содержимого регистров памяти. Эта информация не разрушается и при замене элементов, но при условии, что на эту операцию потрачено не более 15 минут.

Для контроля ввода и индикации результатов применяется "бегущая строка" длиной 63 символа, из которых одновременно индицируется 12. Движение информации в строке происходит автоматически. Им можно управлять с помощью клавиш курсора.

ОСНОВНЫЕ ТЕХНИЧЕСКИЕ ДАННЫЕ

Язык программирования	Бейсик
Система счисления при вводе и выводе чисел	десятичная
Пределы, в которых могут лежат вводимые и выводимые числа	$\pm10^{-4095}...\pm9{,}999999999\times10^{4094}$
Энергонезависимая память: адресных регистров/шагов программы	МК-85 26/1221 МК-85м 26/5317 с возможностью увеличения числа адресных регистров за счет программного ОЗУ по цене 8 шагов за регистр
Индикатор	ЖКИ матричный 12-разрядный с матрицей 5x7
Габаритные размеры, мм	13x166x73
Масса, не более кг	0,15
Цена,руб.	
МК-85	270
МК-85м	540

РЕЖИМЫ РАБОТЫ

При включении питания на бегущей строке появляется надпись: "READY P0", а над ней — "RUN" и "DEG". Компьютер готов к работе в калькуляторном режиме или с программой, записанной в нулевом файле (режим RUN). Надпись DEG означает, что если предполагается работа с тригонометрическими функциями, то их аргумент следует выражать в градусах. Выбор режима производится нажатием клавиши MODE (не путать с клавишей MODE, совмещенной с буквой W) и одной из клавиш 0,1...6 или "точка". Если компьютер находится в режиме программирования, то команда MODE 0 переводит его в режим вычислений.

MODE 1 — режим ввода, проверки и редактирования программы (режим WRT). В этом режиме на индикаторе P 0123456789, а над бегущей строкой вместо RUN — WRT. Цифры означают номера файлов, в которых содержатся программы. Если какой-то файл уже занят программой, то на месте цифры-номера файла будет отображен черный ромбик. Курсор находится под номером того файла, с которым сейчас можно работать. Если же нужно работать с другим файлом, то курсор перемещают на соответствующее место. Для этого нажимают префиксную клавишу S (красное S), находящуюся над клавиатурой, и клавишу цифры, соответствующую номеру файла, с которым пред-

1 *Здесь и далее цены по состоянию на 1 декабря 1991 г.*

стоит работать. Нажатием этих же клавиш запускают программу, записанную в файле, на счет, но только предварительно перейдя в режим MODE 0.

MODE 2 — режим отладки программы по строкам (режим TRACE). Для вхождения в этот режим предварительно переходят в режим MODE 0. О том, что компьютер в режиме TRACE, говорят буквы TR, появляющиеся над бегущей строкой. При запуске программы на бегущей строке отображена информация "Pn — N", где n — номер файла, с которым ведется работа, а N — номер отрабатываемой строки. При каждом нажатии на клавишу EXE отрабатывается один оператор, после чего выполнение программы приостанавливается, о чем свидетельствует надпись STOP, появляющаяся над бегущей строкой. В это время можно проводить проверку переменных, вызывая их на индикатор. Для отмены этого режима нажимают клавиши MODE 3. Буквы TR при этом исчезают.

MODE 4 (5 или 6) указывают, в каких величинах должен быть представлен аргумент тригонометрических функций. Об этом напоминают и буквы над бегущей строкой (DEG — градусы, RAD — радианы, GPA — грады). В программах этот режим задается с помощью клавиш: MODE, совмещенной с буквой W и соответствующей цифры (4, 5 или 6).

MODE "ТОЧКА" — РЕЖИМ РАСШИРЕНИЯ (EXT). В этом режиме на бегущую строку выводятся строчные буквы русского алфавита. Если сначала нажать клавишу MODE, а потом клавишу буквы, то будет выведена соответствующая прописная буква. При нажатии префиксной клавиши S на строку будут выводиться символы, указанные на накладке к клавиатуре, а если нажать префиксную клавишу F — прописные буквы латинского алфавита. Их можно вывести и не используя режим расширения, нажимая клавишу MODE перед нажатием соответствующей буквенной клавиши. О том, что включен режим расширения, говорит надпись EXT над бегущей строкой. Отмена режима расширения происходит при повторном нажатии клавиш MODE и "точка". Надпись EXT при этом гаснет.

ВЕРСИЯ БЕЙСИКА ДЛЯ МК-85

Версия Бейсика, используемая в МК-85, включает в себя 52 ключевых слова и 192 символа, для кодирования которых применяется код ASCI. Коды от 0 до 31 интерпретированы разработчиком и представляют собою различные символы, используемые в режиме расширения. Коды от 32 до 122, как это принято и в других версиях Бейсика, отведены строчным и прописным буквам латинского алфавита, знакам препинания и знакам математических действий. Остальные коды использованы для строчных и прописных букв русского алфавита и некоторых символов. Попутно заметим, что в версии Бейсика, соответствующей ГОСТ 27787-88, содержится 111 ключевых слов, в версии для Спектрума — 90, для Корвета 131 и для PC IBM — 155. Так как значения ключевых слов, за редким исключением, совпадают, то знакомые хоть с одной из версий без труда освоят и эту, а нажав клавиши W,H,O и EXE, узнают и имя автора, который позаботился о том, чтобы сделать это можно было в любом режиме (RUN, WRT или EXT), но забыл включить это слово в Руководство.

Но тем, кто не знаком с языками высокого уровня, придется учить Бейсик. Учебных пособий по этой версии еще нет и статья их не заменит, поэтому многое придется постигать самостоятельно. Тем, кто купил компьютер с библиотекой программ, повезло. Ведь п.3 комплекта поставки, с которого начинается Руководство, гласит: **"Библиотека программ (допускается не комплектовать) 1 шт."**, а мы все знаем, что означает такая запись. Но переживать очень не стоит. Библиотека содержит 59 программ для решения математических, статистических, механических и электротехнических задач. Автору кажется, что она не совсем соответствует назначению справочной литературы, прилагаемой к компьютеру. Основная задача такого пособия — показать, как пользоваться теми или иными конструкциями языка, а вот ее-то библиотека и не решает. Более трети имеющихся в языке ключевых слов в ней не используется. Причем именно тех, которые вызывают у начинающих наибольшие трудности. Можно предположить, что некоторые программы на МК-85 не проверялись. Те, кто захочет воспользоваться программой 1.8 "Вычисление площади треугольника", при пуске программы увидят на индикаторе надпись: "2 угла Y/N?", а должно быть: "сторона, 2 угла Y/N?". Следующий запрос должен быть: "2 стороны и угол Y/N?", а на индикаторе только конец этого сообщения. Дело в

том, что на индикаторе можно прочитать только 11 последних знаков, входящих в текст аргумента оператора INPUT, а если их больше, то даже в нормальном режиме первые знаки пролетают по строке настолько быстро, что прочесть их невозможно, а в режиме быстродействия даже не улавливается то обстоятельство, что начало вопроса "съедено". Некоторые программы содержат ошибки (1.18 и др.).

Сосредоточим основное внимание на составлении, редактировании и текстировании программ. Документированиию любители не придают большого значения, записывая свои программы на клочках бумаги, без комментариев, а иногда и составляя их "живьем", т.е. сразу вводя в ОЗУ без записи. Но кто из нас, программирующих таким способом, не клял себя последними словами, когда нужная программа терялась или ее нужно было применить для другой задачи, а как она делалась уже забылось, и все приходилось начинать сначала. Поэтому, даже начиная осваивать компьютер, не ленитесь записывать. Это поможет не только при приобретении навыков работы, но и сослужит хорошую службу в будущем. Записи нужно делать аккуратно и хранить таким образом, чтобы легко их было найти.

Будем считать, что читатель знает, что означают все слова языка, помещенные в разделе 6.8 руководства, а если не знает, то, перевернув несколько страниц, найдет там список ключевых слов этой версии. Первое, что нужно усвоить, это приоритеты выполнения операций. В первую очередь выполняются действия в скобках. Скобки могут быть многоуровневыми. Например, при вычислении 3310 знаков числа e (основание натуральных логарифмов) используется формула e=2+1/2(1+1/3(1+...+1/1243(1+1/1244)...). В этом и подобных случаях вычисления начинаются с самых внутренних скобок. Следующим приоритетом после скобок идет вычисление функций, потом возведение в степень, за ним умножение и деление, и наконец, сложение и вычитание.

Для изменения естественного порядка выполнения программы служат операторы безусловного (GOTO) и условного (IF <условие> THEN или IF <условие>;) переходов, перехода к подпрограмме (GOSUB) и возврата из подпрограммы к основной программе (RETURN). Все они, кроме последнего, дополняются адресом строки, на которую передается управление, а в операторе условного перехода с разделителем ";" вместо адреса ставится другой оператор. Адрес может быть представлен просто как номер строки, как переменная или математическое выражение, точно соответствующее реально существующему номеру строки, иначе при отработке оператора появится сигнал ошибки. Возможна адресация к другому файлу. В этом случае после операторов GOTO , GOSUB или THEN пишут #n, N. Здесь n — номер файла, в котором находится строка, на которую передается управление, а N — номер этой строки. В том случае, когда управление передается на начальную строку файла, номер строки и разделитель можно не ставить. Например:IF <условие> THEN#3:. Если условие выполняется, то управление будет передано на начальную строку программы, записанной в третьем файле, а если нет, то будет выполняться оператор, стоящий за разделительным двоеточием.

Для повторения отдельных частей программы могут быть организованы циклы. Это делается с помощью конструкции FOR (управляющая переменная), TO (конечное значение управляющей переменной) STEP (знак и величина шага). Далее идет периодически повторяющаяся часть программы, которая называется телом цикла, после чего ставится оператор NEXT и обязательно имя управляющей переменной. При положительном шаге знак "+" не ставят, а если шаг равен единице, то опускают весь оператор STEP. Управляющая переменная, ее конечное значение и шаг могут задаваться числами, переменными или математическими выражениями. Допускается вложение одного цикла в другой (не более четырех) и выход из тела цикла с помощью операторов условного перехода. Войти же в тело цикла можно только с самого начала или вернуться из подпрограммы, если оператор GOSUB находится в теле цикла.

ПРО ТЕБЯ БАСНЯ

Вся информация о текущем состоянии микрокомпьютера находится на индикаторе. Она показана на рис.1. 12 позиций, имеющих номера от нуля до 11, отведены бегущей строке. Над ней служебная информация. EXT, RUN, WRT, DEG, RAD, GPA, TR — говорят о включенном режиме, F и S — о нажатии на одну из префиксных клавиш или клавишу

MODE. В режиме ввода и отладки программ показывается число шагов, оставшихся свободными в программной памяти, которое меняется при вводе или стирании каждой строки. О том, что идет отработка программы, говорит буква P (работа), появляющаяся на месте последней цифры в числе шагов, а о том, что выполнение программы приостановлено, — исчезновение этой буквы и появление надписи STOP. Если на индикаторе нет ни P, ни STOP, то при надписи RUN он находится в режиме обычного калькулятора с алгебраической логикой. Начнем свое знакомство с этого режима.

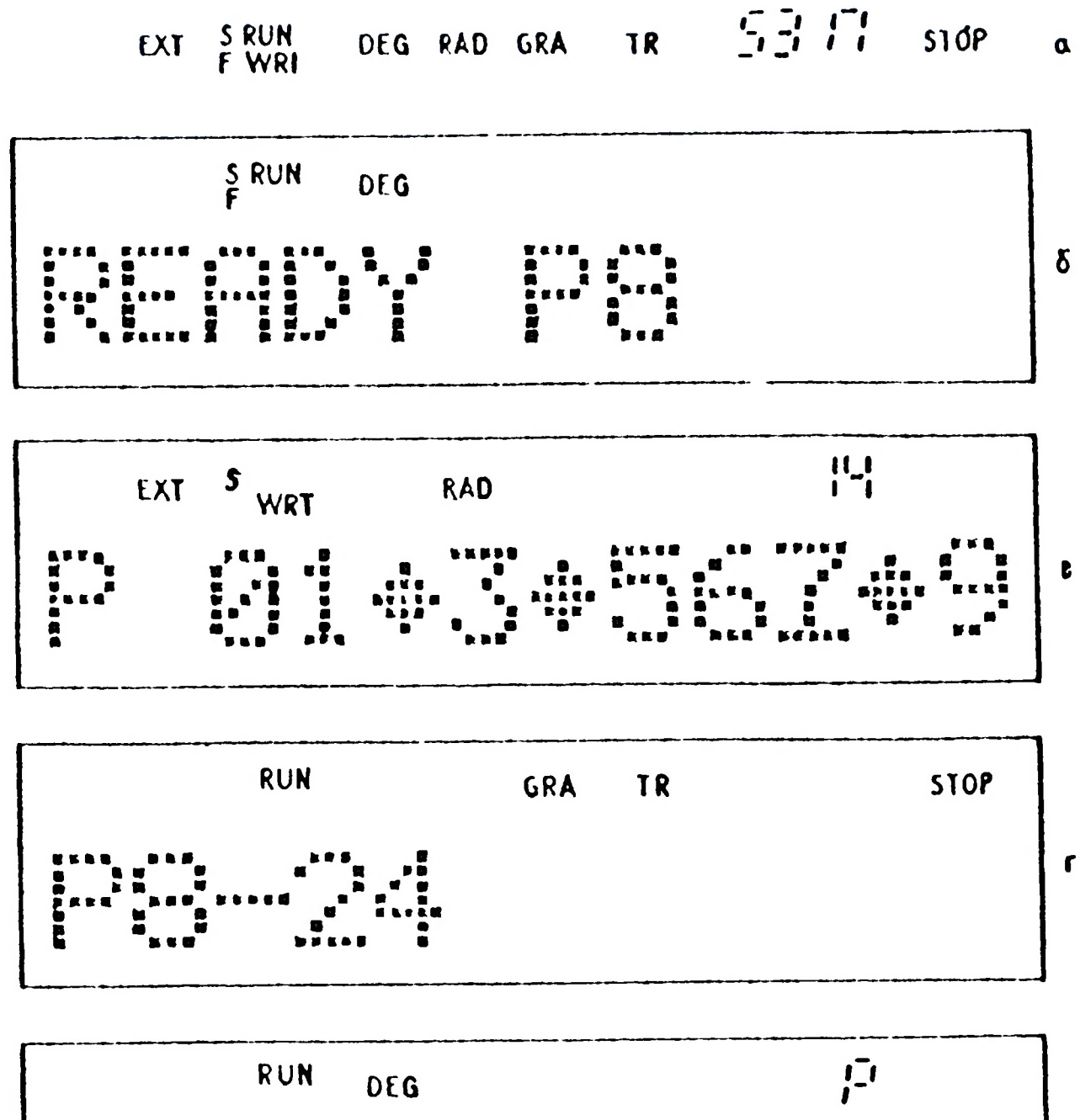


Рис.1. Информация, выводимая на индикатор: а — типы сообщений; б — режим вычислений; в — режим ввода и редактирования программ; г — останов программы командой STOP; д — идет выполнение программы

Попробуем извлечь корень нечетной степени из отрицательного числа. Из школьного курса алгебры известно, что это вполне возможно. $\sqrt[3]{-8} = -2$. Проделаем операции: -8 ↑ (1/3) EXE — и на индикаторе сообщение ERR3, означающее попытку выполнения некорректной операции. Оказывается, МК-85 не умеет возводить в дробную степень отрицательных чисел. Попробуем создать программу, лишенную этого недостатка. Известно, что при извлечении корня из отрицательных чисел получается комплексное число. Возьмем формулу для возведения в дробную степень комплексного числа $Z^p=(a+jb)^p$ и, положив b=0, получим: $a^p=A+jB$, где $A=|a|^p \cos(p(ACS\ (a/|a|)+2k\pi))$; $B=|a|^p SIN(p(ACS(a/|a|)+2k\pi))$: p=m/n: k=0,1...(n-1). Эти соотношения напоминают нам, что корней может быть n и что при их вычислении меняется только величина k, а значит, можно использовать цикл. Формулы универсальны, они позволяют проводить как возведение в степень, так и извлечение корня. Однако есть и ограничения: m и n — положительные целые числа и, кроме того, $n \neq 0$.

Вот теперь, кажется есть все, что необходимо для составления программы. Начнем с перехода в режим WRT. В любой версии Бейсика, если нет подпрограмм, безусловных и условных переходов, производится переход от строки с меньшим номером к ближайшей строке с большим номером. Поэтому строки нумеруются с определенным шагом. Это позволяет при редактировании и коррекции программ вносить поправки "малой кровью",присваивая вновь вводимым строкам номера, лежащие между уже существующими. Для автоматической нумерации строк используется оператор AUTO, после которого пишут заданный шаг для нумерации строк. При отсутствии цифр шага он "по умолчанию" равен 10. Сначала заголовок.

Программа 1. *Возведение в дробную степень отрицательного числа*

10 PRINT "A ↑ (m/n)=A+jB": MODE 5 в наших формулах есть π и запись будет короче, если аргумент тригонометрических функций выражать в радианах.

20 INPUT "a=", Z, "m=", M, "n=", N: P=M/N. В аргументе оператора INPUT в кавычках стоят символы, которые будут выведены на строку вместе с вопросительным знаком. Переменным Z, M и N соответственно присваиваются введенные значения.

30 D=ABS Z ↑P. Вот здесь скобки не нужны, так как в степень возводится модуль, а приоритет функции (здесь — ABS) выше приоритета степени.

40 FOR C=0 TO N-1. Вход в цикл для получения n корней.

50 F=ACS (Z/ABS Z)+2 * C * π

60 PRINT "A";C+1;"=";A=D*COS(P*F),
"B";C+1;"=";B=D*SIN(P*F)

При отработке этой строки на индикатор будет выведена буква A с соответствующим индексом, знак равенства и после небольшой задержки — значение действительной части корня. Останова не будет, так как все эти элементы разделены точкой с запятой. А вот далее стоит запятая, вызывающая останов программы, о чем свидетельствует слово STOP, появляющееся над бегущей строкой. Для появления мнимой части корня нужно вновь запустить программу на счет, нажав клавишу EXE.

70 NEXT C:PRINT "КОРНИ ВСЕ": GOTO 20

Цикл будет отработан n раз. Причем в первый раз управляющая переменная C=0, во второй C=1 и т.д. Цикл будет продолжаться, пока не будет выполнено условие C=N-1. Значение управляющей переменной выбрано так, чтобы оно совпало со значением k в формуле для вычисления корней. [1]

Испытаем нашу программу на том же примере $-\sqrt[3]{-8} = -2$. После пуска программы на счет получим A1=1 B1=1,732050808, A2=-2, B2=0, A3=1 и B3=-B1.

Во многих случаях интерес представляют только действительные корни, а когда они мнимые, то говорят, что корней нет. Кроме того, следовало бы уберечь пользователя от случайных и преднамеренных ошибок при вводе исходных данных, т.е. создать имеющуюся во многих программах систему foolproof (дословно — защита от дурака). Предложим следующую процедуру: после ввода m и n они проверятся и при несоблюдении введенных ограничений пользователю предлагается повторить ввод. На стрску будем выводить только действительную часть того корня, у которого мнимая часть равна нулю. И наконец, если действительных корней нет, то об этом выводится сообщение: "Корней нет". Не хочется бросать уже созданную программу и начинать все сначала, поэтому постараемся использовать все то, что уже сделано. Именно так создавалась программа 2. Из строки 10 с помощью курсора и клавиши DEL были удалены знаки+jB. Напомним, что после "ремонта" строки для ее записи в память нужно нажать клавишу EXE. В строке 20 был изменен номер на 12, и с помощью курсора и той же клавиши DEL была убрана вся информация, стоящая после переменной Z. Но строка 20 осталась в памяти в своем первозданном виде. Для уничтожения строки надо либо под ее номером набрать новую, либо просто набрать ее номер и нажать EXE. Вызвав снова на индикатор строку 20, присвоим ей номер 22 и уберем из нее все, кроме P=M/N. Строки 14, 16, 18 и 20 созданы заново. Для нашего компьютера важен только номер строки, а порядок записи строк в память не имеет значения. При модернизации программы строка 22 была записана в память раньше, чем строки 14 — 20, обеспечивающие работу системы follproof. однако на выполнение программы это не повлияет. Строки 30, 40 и 50 не изменялись. В строке 60 остались только операторы для вывода на индикатор действительной части корня. После удаления всего лишнего командой INS вначале строки было освобождено место для оператора условного перехода, выводящего на индикатор действительную часть корня в том случае, когда мнимая равна нулю. При этом обеспечивается выход из цикла до его завершения, о котором речь шла выше. И наконец, в строке 70 изменено сообщение, выводимое на индикатор.

Программа 2. *Извлечение корня дробной степени из отрицательного числа с элементами системы foolproof*

```
10 PRINT "a^(m/n):MODE 5
12 INPUT "a=", Z
14 INPUT "m=",M:IF M=0;IF FRAC M=0
   THEN 18
16 PRINT "еще раз "; : GOTO 14
18 INPUT "n=", N: IF N <M=> 0 THEN 20:
   IF FRAC N=0 THEN 22
 20 PRINT "еще раз ";:GOTO 18
22 P = M / N
30 D=ABS Z^P30 D=ABS Z^P
```

1 *Автором программ, снабженных порядковыми номерами, кроме специально оговоренных в тексте, является В.А.Хмелюк.*

```
40 FOR C=0 TO N-1
45 IF Z=0;Z=1
50 F=ACS (Z/ABS Z)+2*C*PI
60 IF SIN (P*F)=0;PRINT "A=";
   F = D * COS(P * F): GOTO 12
70 NEXT C:PRINT "корней нет":GOTO 12
```

В этой программе в строках 16 и 18 после выводимого сообщения стоят сразу два разделителя ";:". Первый поставлен для того, чтобы после отработки оператора PRINT не было останова, а второй сообщает, что область действия этого оператора закончилась. Это сочетание разделителей, вместе со стоящим после них оператором GOTO, позволяет создавать сообщение "еще раз m?" и "еще раз n?". Чтобы буквы m и n не сливались с предыдущим текстом, в конце его поставлен пробел.

Вводя дробные или отрицательные значения m или n, а также n=0, убеждаемся, что наша система foolproof работает. При извлечении из отрицательных чисел корней четной степени получаем сообщение "корней нет". Особо нужно отметить, как в тексте программы появилась строка 45, которой сначала не было. Неписаные заповеди программиста говорят о том, что тест позволяет проверить программу при определенных условиях, но не может дать гарантии, что программа безошибочно работает во всем диапазоне допустимых значений аргументов. Эта заповедь нашла отражение и при проверке программы 1. Когда была сделана попытка решить задачу 0^0, на индикаторе появилось сообщение ERR! P0-50. Это означало, что в строке 50 нулевого файла, в котором была записана эта программа, есть ошибка. Для ее локализации нажимают клавишу ANS, которая в обычных условиях служит для вызова предыдущего результата, а при появлении сигнала ошибки переводит компьютер в режим программирования, на индикаторе появляется строка с ошибкой, а непосредственно под ошибочным знаком мерцает курсор. В списке сообщений об ошибках, приведенном в руководстве, ERR! нет, и нам еще предстоит установить тип этой ошибки. При нажатии клавиши ANS на индикаторе появилась строка 50 с мерцающим курсором, находящимся под номером строки. Но оплошность была найдена довольно легко. В аргументе косинуса при a=0 возникает неопределенность типа 0/0. Для ее раскрытия и появилась строка 45. Вот теперь программа работает безупречно.

ОДНОМЕРНЫЙ МАССИВ, ТЕКСТОВАЯ ПЕРЕМЕННАЯ, ВЛОЖЕННЫЕ ЦИКЛЫ, ОПЕРАТОРЫ DEFM, CSR, RUN и VAC

Для открытия дополнительных регистров памяти используется оператор DEFM N, где N — число дополнительно открываемых регистров. О том, что они открыты, говорит сообщение "*** VAR:(26+N)". Для возврата к первоначальному состоянию используют оператор DEFM 0. При этом индицируется сообщение "*** VAR: 26".

Память МК-85 организована следующим образом:

```
A=A(0)
B=A(1)=B(0)
C=A(2)=B(1)=C(0)
.....
Z=A(25)=B(24)=C(23) =.....=Y(1)=Z(0)
----------------------------------------
DEFM 1 A(26)=B(25)=C(24) =.....=Y(2)=Z(1)
DEFM 10 A=(35) B=(34)=C(33) =.....=Y(11)=Z(10)
.....
```

Вместо обычных переменных можно использовать текстовые A$(N)-Z$(N)

При такой организации легко создаются одномерные массивы. Примером организации таких массивов является программа 3 — "Записная книжка", в которой использовано два одномерных массива, составляющих бинарное дерево. Каждому элементу одного массива однозначно соответствует элемент другого. Записная книжка — предмет длительного хранения, поэтому оба массива целесообразно хранить в дополнительных регистрах памяти, а основные регистры использовать для текущей работы. При длине списка M потребуется открыть 2M дополнительных регистра. Программа со-

стоит из двух частей: заполняющей книжку и читающей ее. Сделаем так, чтобы элементы списка имен записывались в нечетные регистры, а их данные хранились бы в соседних четных регистрах. Это позволяет создать достаточно простую процедуру заполнения и чтения, а также замены данных и изменения объема.

Программа 3. *Записная книжка*

```
10 PRINT"Зап.книжка":INPUT
"читать",A$
!RUN 100-ввод списков
20 $="д":IF $<>A$ THEN 60
30 FOR Z=1 TO 100 STEP 2:CSR,3,:
PRINT Z$(Z);:FOR B=0 TO 25
40 NEXT B:NEXT Z
50 CSR,5;PRINT"все"
60 CSR 5:INPUT "имя",A$:FOR Z=1
TO 100 STEP 2
70 IF A$=Z$(Z) THEN 90

80 NEXT Z:PRINT"в списке нет":GOTO 60
90 PRINT Z$(Z+1): GOTO 60
100 FOR Z=1 TO 100 STEP 2
110 INPUT "NAME", Z$(Z)
120 INPUT "DATA", Z$(Z+1): NEXT Z:
GOTO 50
```

При заполнении книжки используются текстовые переменные. Напомним, что в МК-85 может быть одна текстовая переменная $ длиной до 30 символов, а длина всех остальных не должна превышать семи символов. Таких текстовых переменных может быть столько, сколько открыто регистров. При написании программы текстовые переменные заключаются в кавычки. Например: A$="Маша", X$(9)="ЭТО %?".

После ввода в память программы уточняют, сколько же дополнительных регистров еще можно открыть. Для этого дополняют программу строкой 130 DEFM,A:PRINT A. После ввода ее в память очищают индикатор командой AC и набирают RUN 130. Теперь при нажатии клавиши EXE микрокомпьютер автоматически перейдет из режима WRT в режим RUN и отработает строку 130. Цифра, появившаяся на индикаторе, показывает, сколько еще можно открыть регистров. Открывались ли дополнительные регистры, проверяется с помощью операторов DEFM EXE. Если они не открыты, то появляется сообщение "***VAR: 26", а если открыты, то вместо числа 26 будет общее число открытых регистров. Открываем регистры DEFM 2*M EXE.

Начнем загрузку списков. Для этого снова используем оператор RUN. Если после него нет номера строки, то отработка файла начинается с самого начала. Загрузку и коррекцию списков производят редко, поэтому она вынесена в конец программы. Набрав RUN100 EXE, получим сообщение "NAME?". Введем имя первого элемента. Теперь после нажатия клавиши EXE на индикаторе — "DATA?". Вводим элемент второго списка (№ телефона или другую информацию, но не превышающую семи символов). О том, что оба списка введены, говорит сообщение "все". В программе 3 каждый список содержит по 50 элементов, но это только иллюстрация. На самом деле максимальная длина списка M=S/16, где S — число свободных шагов программы после ввода в память программы 3.

Запуск программы в режиме чтения записной книжки производится так же, как и любой другой. Сначала появляется заголовок — "Зап.книжка". После нажатия EXE — вопрос "читать?". Здесь нужно перевести компьютер в режим расширения (EXT) и нажать клавишу "д", если ответ утвердителен, или любую другую, когда он отрицательный, и снова запустить программу на счет. При положительном ответе на индикаторе будут появляться элементы первого списка. Порядок их появления задается переменной Z, а время индикации циклом B, вложенным в цикл Z. Если в цикле B изменить число, стоящее после TO, то изменится и время индикации. Если перейти в режим быстродействия, то время индикации каждого элемента уменьшится. О том, что чтение списка закончено, говорит сообщение "все". Последующее нажатие на EXE приведет к появлению на индикаторе надписи "имя?". Набирая в ответ имя из списка и запуская вновь программу, придем к очередному циклу, в котором введенное имя сравнивается со всеми остальными, имеющимися в списке. При совпадении имен происходит досрочный выход из цикла, и на бегущую строку выводится элемент второго списка, записанный в регистре Z+1. Во всех циклах управляющая переменная Z определяет и элементы списка, первого —как Z$(Z) и второго — как Z$(Z+1).

Использование двух разделителей после оператора PRINT уже знакомо по пред-

ыдущей программе. Оператор CSR n в общем случае определяет позицию на бегущей строке, с которой начинается вывод информации. Запятые показывают, с какой стороны от выводимой информации очищается индикатор. Как и в операторе DEFM и многих других, n может быть числом, переменной или математическим выражением. В этом случае в расчет принимается только целая часть. Если нужно очистить весь индикатор, то применяют оператор CSR, ноль после запятой можно не ставить.

Для изменения длины списка в операторе цикла Z (строки 30, 60 и 100) меняют конечное значение управляющей переменной и делают его равным удвоенной длине списка.

Замена данных внутри списка производится измененем управляющей переменной в строке 100, которой придается такое значение, чтобы при отработке этой строки сразу же осуществился выход на регистр, где находится заменяемая величина.

Ввод списков, в особенности больших, является рутинной работой, занимающей много времени. И бывает очень досадно, когда списки портятся и их приходится вводить заново, а ведь в МК-85 в списке может быть до 500 элементов, а в МК-85м еще больше. Основную опасность для списка представляет оператор VAC, производящий очистку всех открытых регистров. В ряде программ его применение не очень оправдано, так как при операции присвоения уничтожается старое значение переменной и ей присваивается новое. Но, конечно, есть программы, в которых много переменных нужно инициализировать нулем. Тут уж без VAC не обойтись. Для того чтобы защитить информацию в дополнительных регистрах, их нужно закрыть оператором DEFM 0 EXE. В этом случае информация в дополнительных регистрах защищена от стирания командой VAC. Перед работой с записной книжкой дополнительные регистры нужно вновь открыть оператором DEFM 2*M EXE. Сделать это вам напомнят сигналы ошибок ERR6 Pn-30 при чтении списка или ERR6 Pn-70, если работа началась, минуя чтение (n — номер файла, в котором записана программа).

Желающие могут ввести операторы DEFM 2*M и DEFM 0 в текст программы, переложив на компьютер заботу о сохранности списков. Поверьте, что это совсем не трудно сделать.

ПСЕВДОГРАФИКА, СПРАЙТЫ, ОПЕРАТОРЫ LETC, CHR, KEY, ASCI, GETC, MID, LEN и VAL

Операторам псевдографики DRAW x, y и DRAWC x, y повезло больше других. Они использовались в самых первых опубликованных программах для МК-85. Первый из них означает, что нужно установить точку с координатами x и y, а второй — стирает точку. На индикаторе находится 12 матриц 7x5, которые предназначены для индикации различных символов. Так как псевдографика может занимать все поле индикатора, то пределы для x и y: $0 \le x \le 59$ и $0 \le y \le 6$. Само по себе использование этих операторов не содержит никаких трудностей, но в сочетании с другими операторами иногда можно получить разные интересные эффекты, применяемые обычно в играх. С некоторыми из них можно познакомиться в программах К.Вяземского, В.Пономаренко и других, опубликованных в этом журнале.

МК-85 позволяет создавать спрайты. Напомним, что спрайтом называется некоторая конфигурация элементов изображения, создаваемая пользователем, которую с помощью команд, предусмотренных в программе, можно перемещать по экрану как единое целое. В МК-85 спрайты создаются в матрице 5x7 (место, занимаемое одним символом на индикаторе). В этой матрице 7 строк, в каждой из которых может быть не более 5 точек. В коде строки, на том месте, где должна быть точка, ставят 1, а отсутствие точки на данном месте отмечают нулем. Коды строк, показанные в табл. 1, повторно кодируются символами, имеющимися на клавишах. Таблица кодов строк позволяет создать текстовую переменную, которая записывается оператором LETC в память как спрайт. Спрайт хранится в памяти до тех пор, пока в нее не будет записан новый. На индикатор спрайт вызывается оператором CHR 96. Операцию вызова можно производить многократно. Работу со спрайтами иллюстрирует программа 4 К.Вяземского "Профессиональный отбор", в которую внесены небольшие редакционные изменения.

Таблица 1

Таблица кодирования строк спрайтов

Код строк	Клавиша	Код строк	Клавиша	Код строк	Клавиша	Код строк	Клавиша
00000	0	01000	8	10000	G	11000	O
00001	1	01001	9	10001	H	11001	P
00010	2	01010	A	10010	I	11010	Q
00011	3	01011	B	10011	J	11011	R
00100	4	01100	C	10100	K	11100	S
00101	5	01101	D	10101	L	11101	T
00110	6	01110	E	10110	M	11110	U
00111	7	01111	F	10111	N	11111	V

Это шуточный тест, имитирующий испытание для поступающего на работу в пивной бар. Во время испытаний нужно принимать от посетителей кружки, которые подаются слева и справа. Кружка считается принятой, если фигурка официанта вовремя повернулась в нужную сторону. Перечень вакантных мест ясен из программы. Для ввода спрайта в память используют операторы LETC (символы из таблицы кодирования строк) и LETC (текстовая переменная). Кроме того, возможны и другие комбинации в аргументе оператора. LETC (текст, переменная+текст, переменная), а также создание текстовой переменной иными способами. Важно, чтобы в ней были только символы из столбца "Клавиша" таблицы кодирования строк и чтобы их было не больше семи. Спрайт начинает прорисовываться с верхней строки, и если в аргументе оператора LETC будет меньше семи символов, то прорисуются верхние строки, соответствующие имеющимся символам, а в остальных строках просто не будет точек.

Оператор KEY позволяет присвоить текстовой переменной значение символа, соответствующее клавише, нажатой в данный момент. При этом отработка программы идет без останова, а символ не выводится на индикатор. Оператор KEY вместе с оператором условного перехода дает возможность пользователю создавать различные управляющие воздействия, влияющие на ход выполнения программы без ее останова. Достаточно часто встречается сочетание A=ASCI KEY.

Оператор ASCI превращает любую переменную в ее код, т.е. в цифру, что позволяет обойтись без текстовых переменных при использовании оператора KEY. Уже известный оператор CHR производит превращение кода в символ, и в этом смысле он обратен по отношению к ASCI. Как уже упоминалось, код 96 зарезервирован за спрайтом пользователя, во всех остальных случаях существует однозначное соответствие межу кодом и символом.

Программа 4. *Профессиональный отбор*

```
5 !профессиональный тест
10 CSR 0,:PRINT "ПИВНОМУ БАРУ";:
   GOSUB 350
20 CSR,1,:PRINT "требуются:";:GOSUB 350
30 CSR,2,:PRINT "грузчик";:GSUB 350:
   CSR,1,:PRINT "уборщица";
40 GOSUB 350
50 CSR 0,:PRINT "посудомойка";:GOSUB 350
60 CSR,2,:PRINT "официант";:GOSUB 350
70 CSR,2,:PRINT "и бармен";:GOSUB 350
80 VAC:U$="0UVVUC":X$="OO8K222":
   Y$="3325888":Z$="002IVUC"
90 $=CHR 96:V$="=":W$="0":CSR,3,
95 PRINT CHR 25;" =:0 ";CHR 26;:
   GOSUB 350
100 CSR,3,:PRINT "начали";:GOSUB 350:
   CSR 0,
110 A=0:B=-5:C=11:D=15
120 LETC "EE4ELAA":GOSUB 250:
   Q$=KEY
```

```
130 IF Q$<>V$ THEN 150
140 LETC X$:GOSUB 250:IF A=5;
   GOSUB 260: GOTO 120
150 IF Q$<>W$ THEN 170
160 LETC Y$:GOSUB 250:IF C=7;
   GOSUB 270: GOTO 120
170 LETC U$
180 IF A=5;CSR 5:LETC Z$:PRINT $;:
   M=M+1: GOSUB 260:GOTO 120
190 CSR A:PRINT "_";$;:A=A+1
   M=M+1: GOSUB 270:GOTO 120
200 B=B+1:IF B>=0;CSR B:PRINT $;:
   IF C+1<=11; PRINT "_";
210 IF C=7;CSR 7:LETC Z$:PRINT $;:
    IF B>0;CSR B-1:PRINT "_";
220 C=C-1:CSR C:PRINT $;:IF
230 D=D-1:IF D<=11;CSR D:PRINT $;:
240 GOTO 120
250 CSR 6:PRINT $;:RETURN
260 A=B:B=A-1-INT(RAN#*5):
   GOSUB 280: CSR 5:PRINT"_";:
   RETURN
270 C=D:D=C+1+INT(RAN#*4):
   GOSUB 280: CSR 7:PRINT"_";:
   RETURN
280 N=N+1:IF N<=15;RETURN
290 IF M<=1;CSR,2,:PRINT
   "БАРМЕН";: GOSUB 350:GOTO 345
300 IF M<M=>2;CSR,2,:PRINT
   "официант";:GOSUB 350:GOTO 345
310 IF M<=4;CSR,1,:PRINT
   "посудомойка";:GOSUB 350:GOTO 345
320 IF M<=6;CSR,2,:PRINT
   "уборщица";:GOSUB 350:GOTO 345
330 IF M<=8;CSR,3,:PRINT
   "грузчик";:GOSUB 350:GOTO 345
340 CSR ,2,:PRINT"ПШЕЛ ВОН!";:
   GOSUB 350:
345 CSR,2,:PRINT"следующий":
   GOTO 10
350 FOR P=0 TO 50:NEXT P:
```

В этой программе оператор KEY производит присвоение текстовой переменной того символа, с помощью которого управляют фигуркой официанта (ноль — направо, а знак равенства — налево). Если поворот произошел в тот момент, когда спрайты официанта и кружки находятся на соседних позициях, кружка считается взятой, а если он запоздал — разбитой. В строке 280 задано число кружек, подаваемых во время испытания, а в строках 290 — 330 — критерии для занятий той или иной должности. Изменяя эти цифры, можно изменять продолжительность игры и ужесточать или ослаблять критерии. В программе есть элементы системы foolproof, не позволяющие заблаговременно осуществлять поворот или удерживать фигурку официанта длительное время в повернутом состоянии.

Своеобразным учебным пособием по использованию большой группы операторов, отсутствующих в программах библиотеки, прилагаемой к МК-85, служит опубликованная в этом же номере программа В.Пономаренко "Тезей". Использовать ее не только приятно, но и поучительно. На практике можно познакомиться с тем, как выделяют символ из текстовой переменной (GETC) или m символов, начиная с n-ного (MID). Как подсчитывается число знаков в текстовой переменной и для чего это делается (LEN). Как превращается текстовая переменная в число (VAL). Конечно, такое превращение имеет смысл только в том случае, когда текстовая переменная состоит из цифр, в противном случае будет сигнал ошибки.

Особо нужно сказать о длине текста, с которым может работать оператор GETC, формат которого GETC ("текст", N). Здесь N — порядковый номер выделяемого символа. Длина текста ограничивается размерами бегущей строки, на которую нужно уместить сам текст, операторы присвоения и GETC (скобки, знак равенства и разделительную запятую). Вместо текста может быть текстовая переменная $, длина которой может достигать тридцати символов, или сумма текстовых переменных. В последнем случае число символов всех переменных не должно превышать 28. При нарушении этого условия на индикаторе появляется уже известный нам, но не описанный в руководстве, сигнал ошибки ERR!

ДЕТАЛИЗАЦИЯ И УТОЧНЕНИЕ ОТДЕЛЬНЫХ ПОЛОЖЕНИЙ "РУКОВОДСТВА ПО ЭКСПЛУАТАЦИИ"

При появлении сигнала ошибки нажатие клавиши ANS приводит к появлению на индикаторе выражения, в котором допущена ошибка. При этом может быть два случая. Ошибка допущена при вычислениях в калькуляторном режиме и ошибка допущена при работе программы. В первом случае на индикаторе будет только сигнал ошибки, а нажатие ANS приведет к появлению на строке ошибочного выражения. Во втором — на индикаторе, кроме сигнала ошибки, будет номер файла и строки, в которой она произошла. При нажатии клавиши ANS эта строка появится на индикаторе. В обоих случаях мерцающий курсор помогает локализовать ошибку. Так как во втором случае нажатие клавиши ANS переводит компьютер из режима RUN в режим WRT, то можно исправить ошибку сразу. Однако когда идет программирование и после записи очередной строки в память появляется сигнал ошибки ERR1, то пользоваться клавишей ANS нельзя, можно испортить другие файлы. В этом случае индикатор очищают командой AC.

При неправильном вводе данных, запрашиваемых оператором INPUT, нельзя очищать индикатор командой AC. Ошибку ввода нужно исправлять с помощью курсора и операторов DEL и INS.

При вводе программы с автоматической нумерацией строк оператором AUTO прекращение ввода для производства других операций, не связанных с вводом и редактированием, дает отбой этому оператору. При необходимости он может быть введен повторно.

Оператор CHR в сочетании с другими можно использовать для генерации любых символов, имеющихся в микрокомпьютере. Например, B$=CHR (65+26*RAN#) является генератором, генерирующим случайным образом буквы латинского алфавита. При этом находящееся в скобках число не округляется, а берется только его целая часть.

Клавишей ввода порядка можно пользоваться только для чисел, а для переменных нельзя. Нельзя чтобы переменная входила в порядок. Комбинации AE5 или 3EA приводят к появлению сигнала ошибки, а если все же нужно изменить порядок, то приходится прибегать к операциям умножения или деления.

Определение целой (INT) и дробной (FRAC) частей отрицательного числа микрокомпьютер производит неправильно (подробнее об этом см. "Функции пользователя!").

При работе с оператором KEY нужно следить, чтобы компьютер находился в нужном режиме (обычном или EXT), иначе управлять с помощью этого оператора нельзя.

Клавиша останова STOP срабатывает не при нажатии на нее, а при отпускании. При этом над бегущей строкой появляется сообщение STOP. Это сообщение может быть вызвано и другими операторами (STOP в программе, PRINT, TRACE), но независимо от причин, по которым оно появилось, после его появления можно вести вычисления в калькуляторном режиме, вызывать на индикатор переменные, переходить в режим программирования или к другому файлу. В том случае, когда после останова управление не передано на другую часть программы, оператор EXE вызовет продолжение выполнения программы с того места, где произошел останов. Останов может быть и без сообщения STOP. В этом случае сегменты последней цифры в группе знаков, индицирующих число оставшихся шагов программы, включены так, чтобы получилась буква P (работа), и пока она на индикаторе, нельзя производить никаких операций, кроме предписанных программой.

При необходимости превратить в число не всю текстовую переменную, а только одну цифру из нее пользуются операторами VAL GETC. Например: A=VAL GETC ("C4EBBAA", 2). После выполнения этого фрагмента переменной A будет присвоено значение 4. Таким же способом можно выделить цифру из текстовой переменной $. Кроме того, из нее можно выделить несколько цифр, стоящих подряд. VAL MID (m,n).

Надеюсь, что читатель сам найдет и другие сочетания операторов, позволяющие добиться необходимого ему эффекта.

Тот, кто внимательно смотрел предыдущие программы, наверное, обратил внимание, что в программе "Записная книжка" начальная строка кончается так: A$! RUN

100 — ввод списков. Однако текст этого сообщения при отработке программы не выводится. Дело в том, что разделитель "!" используется для того, чтобы отделить программу от комментариев. Конечно, при ограниченном объеме памяти иногда жалко тратить место на комментарий, но указать на такую возможность было бы нужно.

Просматривая таблицу кодов символов (таблица 6.9 Руководства), убеждаемся в том, что символы, закодированные цифрами 0 — 10, отсутствуют на клавиатуре и на накладке к ней, на которой помещены символы, создаваемые в режиме EXT. Их можно вызвать на индикатор только с помощью оператора CHR n, предписывая n значения от 1 до 10. Понятно, что придавая n и другие значения, можно вызвать на индикацию все символы, но остальные можно вызвать и с помощью клавиш, а первые десять нельзя. Чуть дальше будет показано, что это утверждение не совсем строгое, но если говорить об использовании символов с кодами 1 — 10 в программах, то иного пути нет.

Командой TEST, как и при нажатии кнопки "начальная установка", уничтожаются абсолютно все программы, хранящиеся в ОЗУ. К сожалению, после некоторых экспериментов вернуть компьютер к жизни можно только этими операциями, хотя во многих случаях помогает простое выключение питания.

Выше уже говорилось о том, что нумерация строк с определенным шагом позволяет вносить в программу коррективы, которые записываются между уже существующими строками. Некоторые начинающие программисты нумеруют строки подряд, начиная с единицы, объясняя это желанием сэкономить шаги, но экономия тут весьма условна. Ведь на адрес уходит всегда два шага независимо от того, единица это или 9999. (Если номер строки больше, то появляется сообщение ERR2.) Таким образом, экономию на нумерации строк делать нельзя. Единственное, на чем тут можно сэкономить, так это на адресовании из глубины программы к ее началу. Каждый символ в тексте программы занимает один шаг. На оператор GOTO 1 требуется 2 шага, а на GOTO 330 — 4. Не исключено, что в каких-то случаях к этому придется прибегнуть, но если говорить о системе, то из-за этой, часто призрачной экономии не стоит лишать себя возможности дальнейшей работы над программой, чего не позволяет делать нумерация с шагом 1.

Теперь давайте немного полистаем Руководство. Не указан формат округления в операторе RND (M, N), где M может быть числом, функцией или выражением, а N — определяет погрешность округления. Запишем на полях руководства: погрешность округления не превышает 10^N: (с. 19. Здесь и далее в скобках страницы Руководства).

Убеждаемся, что оператор SET n действует либо до повторного введения такого же оператора, но с другим аргументом, либо до выключения питания. При включении формат этого оператора — SET 10. Как и во многих других случаях, n может быть числом, переменной, выражением или функцией (с. 19 и 39).

Исправим опечатку, о которой говорилось раньше. После разделителя ";" переход при невыполнении условия происходит не к следующему по порядку оператору, который может стоять в этой же строке, а к следующей строке. После оператора THEN, кроме указанного формата адресов, может стоять адрес передачи управления на строку другого файла (с. 32).

В примерах циклов на с. 35, в которых конечное значение управляющей переменной задано тригонометрическими функциями sin и cos, цикл будет работать при ряде ограничений, связанных с тем, что конечное значение управляющей переменной по модулю всегда меньше или равно 1. Следовательно, и шаг должен быть меньше единицы. Эти примеры следовало бы прокомментировать или придумать что-то попроще.

У оператора CSR возможен формат CSR, n, — очищающий индикатор слева и справа от текста, вывод которого начинается с позиции n (с. 38).

Возможны следующие форматы оператора LEN. LEN "текст", LEN A$, LEN $ (когда аргумент состоит из одной переменной, его не обязательно заключать в скобки), LEN (A$+B$+...). "Сшивать" переменные можно и в аргументе оператора, поэтому строку 30 (с. 39) можно исключить, а следующую записать так:

```
40 D=LEN (A$+B$)
```

Чтобы не повторяться, укажем, что примерно таким же является формат и других операторов — GETC, LETC, но там нужно учитывать особенности. Например, в LETC число символов, входящих в текст, переменную или их сумму, не должно превышать семи. Все, что говорилось о формате, относится и к оператору VAL (с. 40), но при его использовании текстовая переменная должна состоять только из цифр, иначе будет сигнал ошибки.

При использовании операторов DRAW и DRAWC, формат которых DRAW x, y и DRAWC x, y, на аргумент наложены ограничения, $0 \le x \le 59$, $0 \le y \le 6$ (с. 45). При выходе аргумента за границы об этом нет сигнала. Программа продолжает работать, естественно, без вывода точек на индикатор. Это явление можно наблюдать в программе В.Воронихина и А.Туманова "Поймай точку" (ВТ №9/1991. — С.33).

При диагностике ошибок возможно сообщение, не указанное в руководстве. ERR! — оно появляется если ошибка в аргументе функции, заключенном в скобки. Но вопрос требует исследований, так как утверждать, что нет других ошибок, вызывающих это сообщение, может только разработчик программного обеспечения.

В Руководстве приведено три знака разделения: ",", ":" и ";". Во многих версиях Бейсика официально существует и комбинация разделителей ";:", стоящая после аргумента оператора PRINT и обеспечивающая вывод его на индикатор без останова. В версии МК-85 эта комбинация разделителей приводит к тому же эффекту, хотя официально не разрешена.

ПРАКТИКА — ЛУЧШИЙ УЧИТЕЛЬ

К одной из главных характеристик любого вычислительного устройства относится быстродействие. Для того чтобы можно было проводить сравнение, создаются программы бенчмарк, которые вводятся в различные ЭВМ, а затем сравниваются по времени их выполнения. Трудно не создание бенчмарка, а поиски уже готовой программы, подходящей для нашего компьютера и для которой известно время работы на других ЭВМ. Именно такой программой является программа 5.

Программа 5. *BENCHMARK*

```
10 PRINT "BENCHMARK"
20 K = 0
30 A = 5: A = 0
40 K = K + 1
50 A = K / 2 * 3 + 4 - 5
60 GOSUB 130
70 FOR L = 1 TO 5
80 M(L) = A
90 NEXT L
100 IF K < 1000 THEN 40
110 PRINT "END"
120 END
130 RETURN
```

На PC IBM она выполняется за 37 с, на Apple — за 46 с, на "Искре-226" — за 49 с, на "Tandy Color" — за 51 с, на "Электронике НЦ-80-20" — за 56 с, на "Epson HX-20" — за 101 с, на СМ-1840 — за 104 с. На микрокомпьютере "Fx-702P" (фирма "Касио") — 20 мин. На калькуляторах получены такие результаты: "Электроника МК-52" — 140 мин и "Электроника БЗ-34" — 165 мин. Интересно, за сколько времени справится с этим тестом наш микрокомпьютер в обычном режиме и в режиме быстродействия?

У микрокомпьютера клавиш много, и всегда есть желающие узнать из чисто детского любопытства, а что будет, если нажать вот эту клавишу? Для таких любителей предназначена программа 6.

Программа 6. *ASCI KEY*

```
10 A=ASCI KEY:IF A=0 THEN 10
20 CSR 5:PRINT CHR A;:GOTO 10
```

При ее включении индикатор чист, но стоит нажать любую клавишу, как на индикаторе появится соответствующий символ. Пробы можно производить в обычном режиме и в режиме EXT, при нажатых и отпущенных префиксных клавишах. А вот и обещанный сюрприз. Нажатие на клавиши EXE, курсора, AC и DEL приводит к

появлению на индикаторе символов с кодами 4 — 8, о которых раньше уже говорилось, что вызвать их на индикацию просто нажатием клавиш нельзя. Оказывается можно, но только при работе специальной программы.

В Руководстве указано, что оператор CSR очищает индикатор и обеспечивает вывод информации с нулевой позиции. Используем эти два положения в простенькой программе.

```
10 PRINT "0123456789AB";:
20 CSR,: PRINT "ку-ку"
```

При проверке в режиме TRACE все получается, как было задумано. P_n-10 0123456789AB P_n-20 ку-ку. Но в режиме счета по программе эффект иной. 56789ABку-ку. Сразу подчеркнем, изготовитель и разработчик не виноваты. Они отвечают за то, что разрешено в документах. В данном случае в "Руководстве по эксплуатации", а там такая комбинация разделителей не предусмотрена. Конечно, было бы хорошо, если бы разработчик эту комбинацию узаконил, но даже если в будущих компьютерах при такой комбинации разделительных знаков появится сигнал ошибки ERR2, нельзя будет предъявить претензий. Хотя от души жаль, что пропадут программы. А получить требуемый эффект все же можно, изменив аргумент оператора CSR 0,.

ФУНКЦИИ ПОЛЬЗОВАТЕЛЯ

Многие пользователи занимаются определенным кругом задач, в которых часто встречаются одни и те же функции. С одной из них мы познакомились, когда учили компьютер извлекать корень дробной степени из отрицательного числа. Определение целой и дробной частей числа делается человеком мгновенно, а для машины пришлось ввести специальные функции INT и FRAC, но и ими она пользуется правильно только в том случае, когда работает с положительными числами. По определению целая часть числа меньше или равна числу, а дробная часть — это разность между числом и его целой частью. Пусть N=-7,3, тогда, по определению INT n=-8. Это ближайшее к "-7,3" меньшее целое число, а FRAC n=-7,3-(-8)=0,7. Проверим на МК-85. INT n=-7,, FRAC n=-0,3.

Этот недостаток устранен в двух следующих программах.

Программа 7. *INT*

```
5 ! INT A
10 PRINT "INT A"
20 INPUT "A=",A:IF A>=0 THEN 40
30 IF FRAC A<>0;A = A - 1
40 PRINT INT A:GOTO 20
```

Программа 8. *FRAC*

```
5 ! FRAC
10 PRINT "FRAC"
20 INPUT "A=",A:IF A>=0 THEN 40
30 A=FRAC A+1
40 PRINT FRAC A:GOTO 20
```

Астрономам, геодезистам часто нужно определять угол по его координатам. Такая же задача встречается при вычислениях с комплексными числами и в некоторых других приложениях. О том, как справляется наш компьютер с этой задачей, видно из табл. 2, в которой приведены результаты вычисления ANT(Y/X) при аргументе в градусах. Они помещены в знаменателе. В числителе приведены истинные значений этой функции. Программа 9 "В любом квадранте" по прямоугольным координатам X и У определит угол А в диапазоне $0 \leq A \leq 2\pi$. Размерность аргумента тригонометрической функции задается пользователем (градусы, радианы или грады).

Таблица 2

Результаты вычисления ANT(Y/X) $\frac{\text{Истинное значение}}{\text{МК–85}}$

Y/X	0	1	-1
0	$\frac{0}{\text{ERR!}}$	$\frac{0}{0}$	$\frac{180}{0}$
1	$\frac{90}{\text{ERR!}}$	$\frac{45}{45}$	$\frac{135}{-45}$
-1	$\frac{270}{\text{ERR!}}$	$\frac{315}{-45}$	$\frac{225}{45}$

Из таблицы видна следующая закономерность: когда арктангенс или У отрицательны, к результату, выдаваемому компьютером, прибавляется 180°, а когда отрицательны оба — 360°. Размерность функции ATN задана оператором MODE. Теперь нужно сделать так, чтобы и добавка была в тех же величинах. Так как функция SGN при аргументе больше нуля дает 1, меньше — "-1", то, используя сочетание двух или трех функций, получим нужный результат (ACS SGN Y или ACS SGN ATN (Y/X)). Теперь если аргумент функции SGN положительный — будем получать ноль, а при отрицательном, в зависимости от режима, 2π или 180° или 200 GRA. Надеюсь, что теперь читатель сам разберется в этой программе и представит себе этапы работы над ней.

Программа 9. *В любом квадранте*

```
10 PRINT "A=ATN(Y/X)"
20 INPUT "X=", X, "Y=", Y:
IF X = 0 THEN 50
30 IF Y = 0 ; A = ACS(SGN(X)):
GOTO 70
40 A = ATN(Y/X)+ACS SGN ATN(Y/X)+
ACS SGN Y:GOTO 70
50 IF Y=0; A=0:GOTO 70
60 A = ACS(X) + ACS SGN(Y)
70 PRINT A: GOTO 20
```

КОГДА ДВОЕ ДЕЛАЮТ ОДНО И ТО ЖЕ — ЭТО УЖЕ НЕ ОДНО И ТО ЖЕ

С ЯЗЫКА НА ЯЗЫК. Перед начинающим осваивать вычислительное устройство, будь то микрокалькулятор или компьютер, всегда встает проблема добывания программ. Но МК-85 — новое устройство, программы к которому почти не публиковались, если не считать тех, что были в прилагаемых к нему Руководстве и библиотеке программ, имеющейся у некоторых счастливчиков. Публикации в журналах "Наука и жизнь" (№ 12/90, с. 128) и "Вычислительная техника и ее применение" (№ 9/91, с. 33), конечно, не могут утолить программного голода. Но существует обширнейшая литература на Бейсике, появившимся в 1965 г. для программистов-непрофессионалов. Однако Бейсик — это не один язык, а группа языков, отличающихся друг от друга почти так же, как и языки других родственных групп (болгарский, белорусский, польский, русский, украинский).

Для того чтобы привести в соответствие программу, написанную на одной из версий Бейсика, версии МК-85, нужно знать обе версии. Тут, как и во всяком переводе, нужно хорошо знать оба языка: и тот, с которого переводят, и тот, на который переводят.

Попробуем привести в систему то, что мы уже знаем о Бейсике МК-85. Программа

имеет строчную конструкцию. Одна строка — одно предложение, начинающееся порядковым номером, за которым следуют выражения, состоящие из переменных, операторов и функций, со своими аргументами. Выражения отделяются друг от друга двоеточием. Программа выполняется в порядке возрастания номеров строк, если самой программой не задано иное.

В алфавит версии МК-85 входят русские и латинские строчные и прописные буквы, цифры 0, 1....9. Знаки арифметичесих операций: " ↑ " — возведение в степень, "*" — умножение, "/" — деление, "+" — сложение, "-" — вычитание. Знаки отношений: "=" — равенство. Этот же знак используется как знак присвоения (пусть А будет равно...) "≠" — не равно, ">" — больше, "<" — меньше, "≥" — больше или равно, "≤" — меньше или равно. В этом тексте, как уже говорилось, используются другие знаки. Напомним, что следующие пары знаков обозначают одно и то же: "≠" и "<>", " ↑ " и "^" (знак возведения в степень "кепка"), "≥" и ">=", "≤" и "<=". Кроме этих знаков в алфавите МК-85 используются разделительные знаки для аргументов операторов и ";", а также восклицательный знак для отделения комментариев от текста программы. В других версиях это делает ключевое слово "REM". Для придания наглядности выводимой информации в алфавит введены специальные знаки и символы.

В программу могут входить числа, численные и текстовые переменные. Их формат был показан ранее, когда речь шла об организации памяти. Напомним, что они обозначаются буквами латинского алфавита, к которым в текстовой переменной справа приписан символ денежной единицы $. Текстовая переменная (называемая в некоторых версиях литерной или строковой) может включать в себя до 7 ($ до 30) любых символов языка, кроме кавычек, в которые она заключается. Если текстовая переменная не содержит символа, то она обозначается двойными кавычками (A$=""). Длина такой переменной равна нулю. Так же как и она на индикаторе выглядит переменная A$="␣" (где знаком "␣" обозначен оператор пробела SPC), но длина этой переменной равна единице.

В нашей версии отсутствуют цельночисленные переменные (A%), переменные удвоенной точности (A#), цельночисленное деление, логические операции, нет оператора размера массива DIM и многого другого. Поэтому при выборе программ для перевода нужно смотреть, а можно ли их перевести. Иногда можно выйти из положения, зная, что один оператор можно заменить группой операторов. Иногда приходится менять алгоритм. Одним словом, перевод с одного языка на другой — работа творческая, а дословный перевод не всегда самый лучший.

В приведенном ниже списке ключевых слов версии для МК-85 применены следующие обозначения:

M, P, R — число или текстовая переменная, или функция, или математическое выражение. В двух последних случаях, если это аргумент функции, то он должен ставиться в скобки.

m, n, N — числа.

Переменные, обозначенные символами A и A$, в программах могут иметь формат A...Z, A(N)...Z(N) и A$...Z$, A$(N)...Z$(N).

Список ключевых слов версии БЕЙСИК МК-85

ABS — абсолютное значение. Формат — ABS M.

AC — очистка индикатора.

ACS — арккосинус. Формат — ACS M.

ANS — в режиме калькулятора вызов на индикатор результата предыдущей операции. При сообщении об ошибке вызов на индикатор выражения, в котором она произошла;
— в режиме ввода и редактирования программы после сообщения об ошибке вызов на индикатор строки с ошибкой;
— при останове в режиме работы по программе в результате ошибки — переход в режим программирования и вызов на индикатор ошибочной строки. Мерцающий курсор ее локализует.Это позволяет сразу внести исправления.

ASCI — код ASCI символа, стоящего после знака функции. Формат — ASCI "любой символ клавиатуры, получаемый при нажатии на клавишу", ASCI A$ (или $). В этом случае текстовая переменная должна содержать только один символ ASCI KEY (см. KEY).

ASN — арксинус. Формат — ASN M.
ATN — арктангенс. Формат — ATN M.
AUTO — автоматическая нумерация строк. Формат — AUTO (шаг 10), AUTO N (шаг N).
CHR — вывод на индикатор символа по его коду (операция, обратная ASCI). Формат — CHR M, где 0<M<191. Код 96 зарезервирован для спрайта пользователя.
CLEAR — очистить файл Pn (см. S) программной памяти.
CLEAR A — очистить все файлы программной памяти.
COS — косинус. Формат COS M.
CSR — очистка индикатора и вывод информации с n-й позиции. Форматы: CSRn — вывод информации начиная с позиции n без очистки индикатора. CSR,n — то же, но с очисткой слева от выводимой информации. CSR n, — то же, но с очисткой справа. CSR,n, — то же с очисткой слева и справа. CSR (CSR 0) — полная очистка и вывод информации с нулевой позиции.
DEFM — открытие и закрытие дополнительных регистров памяти за счет ОЗУ программ, по цене 8 шагов за регистр. Форматы: DEFM N EXE — открыть N дополнительных регистров памяти. Формат сообщения об исполнении: "***VAR: n", где n=N+26. DEFM 0 EXE — закрыть дополнительно открытые регистры. Формат сообщения об исполнении: "*** VAR: 26". DEFM EXE — сколько открыто дополнительных регистров? Ответ: "*** VAR: n". Открыто (N=n-26) дополнительных регистров. Запрос в режиме счета по программе. Сколько еще можно открыть дополнительных регистров при данном заполнении ОЗУ программ? 10 DEFM, A:PRINT A. Еще можно открыть A дополнительных регистров.
DEG — аргумент тригонометрических функций в градусах.
DEL — стереть символ, под которым находится курсор.
DRAW — формат DRAW x,y — установить на индикаторе точку с координатами x и y.
DRAWC — формат DRAWC x,y — стереть точку с координатами x и y. В Операторах псевдографики x и y могут быть величины, описываемые литерой M и не должны выходить из пределов $0 \le x \le 59$ и $0 \le y \le 6$.
EE — ввод порядка для числа в показательной форме. Формат nEN или nE-N (n и N вещественные числа, а N - только целое). При отрицательном N на индикаторе nĒN.
END — конец программы.
ERR1 — ошибка в использовании памяти. Формат — ERR1 Pn-N (при ошибках, совершенных в калькуляторном режиме, отсутствуют указания на номер файла (Pn) и строки (N), в которых произошла ошибка).
ERR2 — синтаксическая ошибка. Формат ERR2 Pn-N.
ERR3 — математическая ошибка. Формат ERR3 Pn-N.
ERR4 — адресация к несуществующему номеру строки операторами GOSUB, GOTO или THEN (см.) Формат ERR4 Pn-N.
ERR5 — ошибка в аргументе оператора. Формат — ERR5 Pn-N.
ERR6 — использование переменной, не зарезервированной оператором DEFM. Использование одной и той же переменной в качестве обычной и текстовой. Формат ERR6 Pn-N.
ERR7 — использование оператора RETURN (см.) без GOSUB или оператора FOR без NEXT (см.). Уровень вложения подпрограмм больше чем 8 или уровень вложения циклов больше чем 4. Формат ERR7 Pn-N.
ERR8 — в обычной текстовой переменной больше чем 7 знаков, в текстовой переменной $ больше чем 30 знаков. Формат ERR8 Pn-N.
ERR! — ошибка в аргументе функции или операторе, заключенном в круглые скобки.
EXE — команда пуска. В калькуляторном режиме она заменяет знак равенства в калькуляторах с алгебраической логикой;
— команда пуска после останова программы операторами PRINT, INPUT или командой STOP;
— команда "конец строки" в режиме программирования, по которой производится ввод программы в память и осуществляется переход к следующей строке.
EXT — сообщение о введении режима расширения.
EXP — возведение числа e в степень M. Формат EXP M.

F — нажата префиксная клавиша F.

FOR — оператор цикла. Формат:

```
FOR A=M TO P STEP R
.....
.....
NEXT A
```

A — управляющая переменная, дающая имя циклу. M — начальное значение управляющей переменной. P — конечное значение управляющей переменной, R — шаг изменения управляющей переменной. При R=1 STEP R опускается. NEXT обеспечивает возврат к оператору FOR до тех пор, пока не будет выполнено условие A=P.

FRAC — выделение дробной части M. Формат FRAC M.

GETC — выделение N-го символа из литерных конструкций. Формат: GETC ("текст", N), GETC (A$, N), GETC ($, N), GETC (A$+B$+...,N). Общее количество символов при конкатенации текстовых переменных не должно превышать 28.

GOSUB — передача управления подпрограмме GOSUB M, записанной в этом же файле, начиная со строки с номером M. GOSUB#n, записанной в другом файле (n), начиная с начальной строки. GOSUB#n,N, записанной в другом файле (n), начиная с N-й строки.

GOTO — безусловный переход. Форматы аналогично GOSUB (см.).

GRA — аргумент тригонометрических функций в градах.

IF — оператор условного перехода. Форматы: IF — (условие) THEN M: При выполнении условия управление передается на строку, номер которой M (форматы адреса такие же, как у GOSUB). В противном случае выполняется оператор, стоящий после двоеточия. IF — (условие); — при выполнении условия управление передается оператору, стоящему после точки с запятой, в противном случае — на следующую строку.

INPUT — оператор управления вводом данных. Если в аргументе оператора нет текста, то на индикаторе "?", в противном случае перед вопросительным знаком — текст. Форматы: INPUT A: INPUT A, B, C: INPUT A$ (или $). INPUT "текст", A (или A$): INPUT "текст", A, "текст", A$ (или $). Возможны и другие комбинации.

INS — ввести свободное место левее курсора (обычно для пропущенного символа).

INT — выполняет те же функции, что и FIX в других версиях, т.е. выделяет цифры, стоящие перед десятичной точкой, сохраняя знак числа. Формат INT M.

KEY — присвоение текстовой переменной имени, соответствующего символу нажатой клавиши. Формат A$=KEY. Присвоение переменной кода, соответствующего символу нажатой клавиши. Формат A=ASCI KEY.

LEN — определение числа символов в литерных конструкциях. Форматы: LEN "текст". LEN A$ (или $), LEN (A$+B$+...) При конкатенации число символов не более 28.

LETC — запись в память спрайта пользователя. Формат аргумента такой же, как у оператора LEN, но все символы должны браться только из таблицы кодирования символов, LEN аргумента LETC≤7 (см. CHR 96).

LIST — в режиме исполнения программы чтение ее, начиная с начала. LIST N со строки N. В режиме ввода и редактирования ВЫЗОВ на индикацию строки с номером N. При отсутствии N — с начальной строки. Форматы LIST или LIST N.

LN — натуральный логарифм. Формат LN M.

LOG — десятичный логарифм. Формат LOC M.

MID — выделение из $ m символов, начиная с n-го, m и n могут выражаться через M. Обязательное условие m+n<LEN $+1.

MODE 0 — команда перехода в режим вычислений или работы с программой. После подачи команды на индикаторе сообщения "READY Pn и "RUN". n — номер файла, с которым можно работать.

MODE 1 — команда перехода в режим ввода и редактирования программы. После подачи команды на индикаторе P 0123456789 и WRT. После P — номера файлов. Если какие-то файлы заняты программой, то вместо номера

— черный ромбик. Мерцающий курсор под номером той программы, которую можно вызвать командой LIST.

MODE 2 — команда перехода в режим отладки программы по строкам. Если на индикаторе сообщение WRT, то перед ней дать команду MODE 0. На индикаторе RUN и TR.

MODE 3 — отмена режима отладки по строкам. Надпись "TR" на индикаторе гаснет.

MODE 4 — аргумент тригонометрических функций в градусах.

MODE 5 — аргумент тригонометрических функций в радианах.

MODE 6 — аргумент тригонометрических функций в градах.

MODE . — команда включения режима расширения EXT. Отмена режима — повторное нажатие тех же клавиш.

MODE — переключение при работе программы. Форматы: MODE 4 (5,6)

NEXT — см. FOR

P — работает программа.

PRINT — вывод на индикатор информации. Формат: PRINT A — вывод значения переменной A. PRINT A$ — вывод значения переменной A$. PRINT "текст", PRINT "текст"; A (или A$); "текст". ,PRINT — "текст", A или (A$), "текст". Возможны и другие варианты. При разделителе ";" — текст выводится без останова, а при разделителе "," после каждого фрагмента нужно нажать клавишу EXE.

RAD`' — аргумент тригонометрических функций в радианах.

RAN# — генерация случайного числа. Формат RAN#.

READY Pn — подготовлена работа с файлом n.

RETURN — возврат из подпрограммы. Передача управления оператору, стоящему после GOSUB.

RND(— округление. Формат — RND (M, N). При N<0 округляется N-й знак после запятой, при $N \geq 0$ -разряд 10^N.

RUN — запуск программы: Форматы: RUN EXE — запустить программу с начальной строки. RUN N EXE — запустить программу с N-й строки.

RUN — установлен режим вычислений в режиме калькулятора или по программе.

Sn — в режиме вычислений запустить программу, хранящуюся в файле n. В режиме ввода и отладки программы перейти на работу с файлом n, формат Sn.

S — начата префиксная клавиша S.

SETM — провести усечение результата, оставив в нем M знаков. $0 \leq M \leq 10$. Действует до подачи команды с другим M или до выключения питания. При включении питания формат — SET 10.

SGN — сигнум-функция. Формат SGN M.

$$\text{SGN M} = \begin{cases} 1 \text{ при } M>0, \\ 0 \text{ при } M=0, \\ -1 \text{ при } M<0 \end{cases}$$

SIN — синус. Формат SIN M.

SPC — установить пробел.

SQR — извлечь квадратный корень. Формат — SQR M.

STEP — см. FOR

STOP — команда останова программы. Исполняется при отпускании клавиши STOP.

STOP — команда останова программы, вставляемая в программу.

STOP — сообщение об останове программы командой (оператором) STOP или оператором PRINT

THEN — см. IF

TO — см. FOR

TAN — тангенс. Формат — TAN M.

TR — включен режим отладки TRACE.

VAC — полная очистка всех открытых регистров памяти. Формат — VAC.

VAL — превращение текстовой переменной в числовую. Имеет смысл только в том случае, когда она состоит из одних цифр. Формат: VAL A$ (или $), VAL (A$+B$+...)

WRT — вход в режим ввода и редактирования программы.

Вот теперь, сравнивая этот список с конструкциями других версий Бейсика, можно решить, какие программы можно взять непосредственно, какие переделать или слегка подправить, а над какими придется трудиться мучительно и долго, без всякой надежды на то, что перевод удастся сделать.

Попросите любого человека быстро назвать часть лица, великого поэта и фрукт. С вероятностью 0.85 незнакомые с этим немудренным психологическим тестом ответят: нос, Пушкин, яблоко, а с вероятностью 0.97 в ответе будет присутствовать хотя бы одно из этих заветных слов. У программиста, приступающего к освоению ЭВМ или нового языка, тоже есть три заветных программы: календарь, угадай число и квадратное уравнение. Так как эти программы можно найти во многих книгах по программированию, то процессы перевода показаны именно на них. На рис.2 приведены программы, взятые из книг В.Ф.Очкова и В.А.Хмелюка "От микрокалькулятора к персональному компьютеру" и В.Ф.Очкова и Ю.В.Пухначева "24 этюда на Бейсике".

Начнем с календаря. Как и при всяком переводе, сначала нужно прочесть и уяснить, о чем идет речь. Программа определяет день недели по введенным порядковым номерам дня (K), месяца (M) и года (G). При введении второй даты определяется для нее день недели и число дней, прошедшее от первой даты до второй. Формат вывода информации такой: для дня недели выводится его код (0 — воскресенье, 1 — понедельник, 2 — вторник, 3 — среда, 4 — четверг, 5 — пятница и 6 — суббота), а разность дат выводится как число.

Нулевой строки в нашей версии быть не может, поэтому присвоим начальной строке порядковый номер 1. При обращении к регистру памяти D (строки 2 и 7) его аргумент нужно поставить в скобки. Попутно исправим опечатку, которая все равно появится при тестировании. В строке 45 вместо 119 должно быть 114. Остальные строки не изменяются.

```
10 A=1000+INT(9000*RND(1))
20 N=0
30 N=N+1:PRINT N"-я попытка";:
   INPUT B
40 IF B>A THEN
            PRINT"перелет"
50 IF B<A THEN
            PRINT"недолет"
60 IFB<>A GOTO 30
70 PRINT"угадали"
```

а)

```
 0 GOSUB 11
 2 D1=D:PRINT N:GOSUB 11
 7 R=D-D1:PRINT N,R:STOP
11 INPUT K,M,G
15 IF M>=3 THEN 26
20 M=M+12:G=G-1
26 M=M+1:C=INT(M*30.6)
35 U=INT(G*365.25)
45 D=C+U+K-119
55 C=INT(D/7):N=D-C*7:RETURN
```

б)

```
10 PRINT"РЕШЕНИЕ КВАДРАТНОГО УРАВНЕНИЯ A*X*X+B*X+C=0
20 INPUT"ВВЕДИТЕ ЧЕРЕЗ ЗАПЯТУЮ ЗНАЧЕНИЯ A,B,C"A,B,C
30 IF A=0 AND B=0 AND C=0THEN 110
40 IF A=0 AND B=0 THEN 120
50 IF A=0 THEN 130
60 IF C=0 THEN 140
70 R=-B/(2*A):D=B*B-4*A*C:I=SQR(ABS(D))/(2*A)
80 IF D<0 THEN 150
90 IF D=0 THEN 160
100 PRINT"КОРНИ ДЕЙСТВИТЕЛЬНЫЕ X1=";R+I;"X2=";R-I:STOP
110 PRINT"КОРНИ МОГУТ ПРИНЯТЬ ЛЮБОЕ ЗНАЧЕНИЕ":STOP
120 PRINT"УРАВНЕНИЕ ВЫРОЖДЕНО":STOP
130 PRINT"КОРЕНЬ ЕДИНСТВЕННЫЙ X=":STOP
140 PRINT"КОРНИ ДЕЙСТВИТЕЛЬНЫЕ X1=0 X2=";-B/A:STOP
150 PRINT"КОРНИ КОМПЛЕКСНЫЕ X1=";R;"+";ABS(I);"*I
    X2=";R"-";ABS(I);"*I":STOP
160 PRINT"КОРНИ СОВПАДАЮТ X1=X2=";R:STOP
```

в)

Рис.2. а — угадай число; б — календарь; в — квадратное уравнение

Программа 10. *Календарь*

```
1 GOSUB 11
2 D(1)=D:PRINT N:GOSUB 11
7 R=D-D(1):PRINT N,R:GOTO 1
11 INPUT K,M,G
15 IF M>=3 THEN 26
20 M=M+12:G=G-1
26 M=M+1:C=INT(M*30.6)
35 U=INT(G*365.25)
45 D=C+U+K-114
55 C=INT(D/7):N=D-C*7:
RETURN
```

Для проверки введем сегодняшнюю дату и получим код дня недели. Для второго теста возьмем даты, незабываемые людьми старшего поколения — 21 июня 1941 г. и 9 мая 1945 г. Первая дата — суббота, вторая — среда. И между ними — 1418 дней и ночей.

В программе "Угадай число" машина с помощью ГСЧ (генератора случайных чисел) "задумывает" число в диапазоне 1000 — 9999, которое нужно отгадать, вводя пробные числа. О соотношении задуманного числа (A) и пробного (B) говорят сообщения "недолет" и "перелет". Машина ведет счет пробным числам, сообщая о нем в диалоговом режиме, при приглашении ввести новое пробное число, в формате "N-я попытка?", где N — порядковый номер вводимого пробного числа. О том, что число угадано, говорит сообщение "угадали".

Вместо оператора RND(1) используем имеющийся в нашей версии оператор RAN#. В этом переводе так сделать можно, но нужно помнить, что однозначного соответствия между этими операторами нет. Во многих версиях Бейсика с помощью RND можно вернуться к началу генерируемой псевдослучайной последовательности, что бывает очень нужно при отладке программы. В МК-85 к началу последовательности можно вернуться, только подав команду команду TEST или нажав кнопку "начальная установка", но при этом будут уничтожены все программы, хранящиеся в ОЗУ пользователя. Возможно, что есть и не такой болезненный способ, но в Руководстве о нем ничего не говорится. Для желающих поэкспериментировать сообщаем первые псевдослучайные числа, с которых начинается последовательность в МК-85: .2113342285, .8715209961, .815032059, .8704633804, .4621887207.

Ряд версий Бейсика позволяет исправить явные промахи программиста, такие, например, как в строке 30, где после N не поставлена точка с запятой. Словом, там, где иного быть не может, машина исправляет ошибки человека, а человек иногда злоупотребляет этим качеством.

Разделитель ";:" нам уже знаком. Здесь он позволяет избежать лишнего нажатия на клавишу EXE. Конструкцию IF B>A THEN PRINT (строка 40) нужно записать как IF B>A; PRINT. Аналогичное изменение нужно сделать и в следующей строке. А строку 60 написать так: IF B=A THEN 30.

Программа 11. *Угадай число*

```
10 A=1000+INT(9000*RND#)
20 N=0
30 N=N+1:PRINT N:"-я попытка";:
INPUT B
40 IF B>A;PRINT "перелет"
50 IF B<A;PRINT "недолет"
60 IF B<>A THEN 30
70 PRINT"угадали":GOTO 10
```

Для тестирования нужно знать A. Однако разделитель ";:" в строке 30 не позволяет это сделать элементарно, поэтому переведем компьютер в режим TRACE и после появления сообщения P-20 узнаем A (A EXE). Теперь выключим режим отладки и нажмем клавишу EXE. При запросе "1-я попытка ?" введем B=A-1. При последующих запросах B=A+1 и B=A. Если реакцией компьютера будут слова: недолет, перелет, угадали, то все ветви программы проверены. Можно играть.

Перейдем к программе решения квадратного уравнения. В строке 10 — заголовок. Он длинный и есть смысл его сократить и даже исключить. Иногда ради уменьшения длины программы это приходится делать. Через запятую вводить переменные в нашей версии нельзя. Их нужно вводить отдельно и по одной. У нас нет оператора AND (логическое И), но если знать, что записи в строках 30 и 40 означают: "если все коэффициенты рав-

ны нулю" или "если A и B равны нулю", то из положения выйти можно. В программе отсутствует условие B=0. Его можно добавить, введя дополнительную строку 65, а для вывода результата придется дописать новую строку в конце программы. Строки 50 — 160 можно не менять, но при этом нужно учесть, что в программу заложен формат вывода информации для дисплея, на котором она будет смотреться (вместо всех переменных выводятся их численные значения). Вот формат вывода, создаваемый:

В строке 100:	корни действительные X1=R+I X2=R-I
В строке 110:	корни могут принять любое значение
В строке 120:	уравнение вырождено
В строке 130:	корень единственный X=-C/B
В строке 140:	корни действительные X1=0 X2=-B/A
В строке 150:	корни комплексные X1+R+ABSC(I) X2=R-ABS(I)
И наконец, в строке 160:	корни совпадают X1=X2=R

Этот формат вывода для бегущей строки, на которую одновременно выводится только 12 символов, очень неудобен. В приведенном ниже варианте программы использованы привычные для автора форматы, но это, конечно, не канонический перевод. Так как обычно приходится решать несколько уравнений с различными коэффициентами, то для удобства повторного запуска оператор STOP в строках 100 — 160 заменен на GOTO 20.

Программа 12. *Решение квадратного уравнения*

```
10 PRINT"кв.ур-ие"
20 INPUT "A=",A,"B=",B,"C=",C
30 IF A=0;IF B=0;IF C=0 THEN 110
40 IF A=0;IF B=0 THEN 120
50 IF A=0 THEN 130
55 IF B=0 THEN 170
60 IF C=0 THEN 140
70R=-B/(2*A):D=B*B-4*A*C:
I=SQR(ABS D)/(2*A))
80 IF D<0 THEN 150
90 IF D=0 THEN 160
100 PRINT"X1=";R+I,
"X2=";R-I:GOTO 20
110 PRINT"любое значение":GOTO 20
120 PRINT "вырожденое":GOTO 20
130 PRINT "X=";-C/B:GOTO 20
140 PRINT "X1=0, X2=";-B/A:GOTO 20
150 PRINT "Re=";R,"Im=";ABS I:
GOTO 20
160 PRINT "X1=X2=";R:GOTO 20
170 PRINT "X1=";R=SQR(C/A),"X2=";-R
180 GOTO 20
```

Знакомые с Бейсиком легко приспособят для МК-85 программу, показанную на рис.3.

Пример 4.7. Программа расчета среднего значения

```
10 DATA 10,56,16,9,23,
   53,41,2,18,13,32
20 READ N
30 S=0
40 K=1
50 READ A
60 S=S+A
70 K=K+1
80 IF K<=N THEN 50
90 S=S/N
100 PRINT 'среднее значение',S
110 END
```

Рис.3. Операторы DATA и READ

Она взята из книги В.Т.Сергованцева и В.В.Бледных "Вычислительная техника в инженерных и экономических расчетах" (с. 103). В аргументе оператора DATA 11 чисел. Первое — количество чисел, берущееся для определения среднего значения, последующие — числа, для которых оно определяется. Оператором READ N переменной N присваивается значение соответствующего числа из аргумента оператора DATA. K — управляющая переменная цикла суммирования, в котором происходит суммирование этой переменной с суммой ранее извлеченных чисел. После окончания цикла вычисляется среднее значение.

В МК-85 нет операторов DATA и READ. Их заменяют так: строку 10 — исключают. Присвоение значения переменной N делают оператором INPUT, а вместо оператора READ A (строка 50) организуется последовательный ручной ввод чисел, среднее значение которых нужно определить. Чтобы не сбиться при вводе на индикатор выводится порядковый номер вводимого числа. Однако сделать это с помощью одного оператора INPUT не удалось и пришлось прибегнуть к помощи оператора PRINT.

Программа 13. *Расчет среднего значения*

```
20 INPUT"N=",N
30 S=0
40 K=K+1
50 PRINT "a";K;"=";:INPUT A
60 S=S+A
70 K=K+1
80 IF K<=N  THEN 50
90 S=S/N
100 PRINT"Scp=";S
110 GOTO 20
```

ОТ МИКРОКАЛЬКУЛЯТОРА К МИКРОКОМПЬЮТЕРУ. До 2 апреля 1991 г. МК-85 стоил 145 р., что по сравнению с программируемым микрокалькулятором "Электроника МК-52" (115 р.) совсем недорого. Поэтому многие владельцы программируемых микрокалькуляторов стали владельцами МК-85. И тут тоже возникла проблема перевода. Жалко отказываться от программ, наработка которых шла годами, но из-за бесскобочной логики их нельзя использовать без существенной переделки. К тому же книги, в которых есть программы для программируемых калькуляторов, издавались массовыми тиражами и там есть ряд программ, которые хотелось бы использовать и на МК-85.

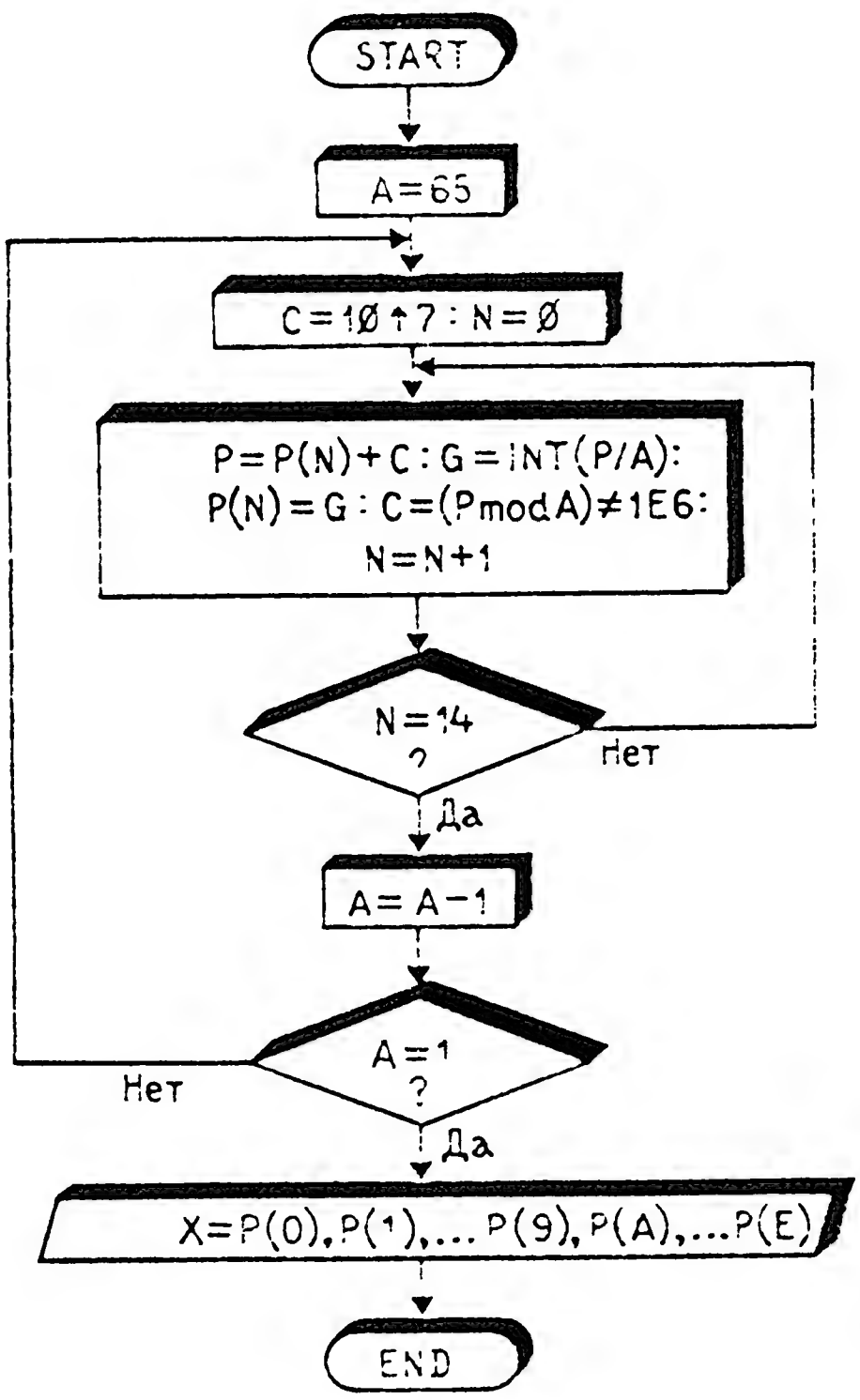


Рис.4. Алгоритм вычисления цифр числа e

Существуют две магистральные линии в осуществлении перевода: перевод описательный и перевод дословный. В первом случае находят алгоритм или математическую основу, составляющую суть программы, и создают новую программу на соответствующем языке. Во втором — создают интерпретатор, интерпретирующий понятия одного языка в понятиях другого.

Примером первого пути является программа 13, вычисляющая 3310 знаков e (основания натуральных логарифмов). Подобная программа была опубликована в журнале "Наука и жизнь", № 6/91, с. 132. На рис.4 показан алгоритм, по которому она работает. Пользуясь им, и была создана программа, учитывающая особенности МК-85. В первом варианте перевода в регистр записывалось по восемь цифр, в остальном алгоритм не менялся. Но потом вспомнилось предложение Б.Ходова о полном использовании емкости регистра микрокалькулятора, храня в нем два числа, разделенные десятичной точкой (Б.Ходов "Если не хватает адресных регистров"// Наука и жизнь, № 10/85, с. 134). Идея оказалась применимой и к микрокомпьютеру, что позволило хранить в одном регистре две группы по пять цифр, обрабатывая каждую отдельно. Эти новшества позволили создать достаточно короткую и эффективную программу.

К тому же подтвердилась одна из неписаных заповедей программиста: программирующий сам наслаждается дважды: когда программирует и когда работает с программой. Результатом стала программа 14.

Программа 14. *Вычисление 3310 знаков*

```
5 PRINT "Число e"
10 VAC: FOR A = 1244 TO 2 STEP-1:
C = 10 ^ 5
20 FOR B = 0 TO 330
30 D = INT(G(B)) + C:
E = INT(D / A): C = D - A * E
40 D = (FRAC G(G(B))+C)*(10 ^ 5):
F = INT(D/A):C=(D-A*F)*(10^5)
50 G(B) = E+F/10^5:NEXT B:NEXT A
60 FOR B = 0 TO 330: PRINT G(B):
NEXT B:PRINT "конец":END
```

Для теста, который в режиме быстродействия отрабатывается за 5 с, A=3, а в строках 20 и 60 FOR B=0 TO 10, после чего в регистрах G (0) — G (10) должны быть цифры 66666.66666. Если тест прошел, то определяют, сколько знаков нужно (или можно) получить. С учетом резерва для переменных A и B программа может занимать в памяти 170 — 180 шагов. При отсутствии других программ это позволяет открыть в МК-85 130 дополнительных регистров памяти, а в МК-85м — 642. Для записи цифр числа e используются и 20 основных регистров — G(0) — G(19), а так как в каждый регистр записывается по 10 цифр, то, пользуясь программой 13 на МК-85, можно вычислить 1500 цифр числа e, стоящих после запятой, а в МК-85м — теоретически 6620. Для обеспечения необходимой точности нужно соответствующим образом выбрать A. Это делается с учетом теоремы об остатке монотонно сходящегося ряда, гласящей, что последний удерживаемый член ряда по модулю превосходит сумму всех отбрасываемых. А так как e=1+1/1!+1/2!+...+1/A!+..., то нужно, чтобы величина 1/A! была бы меньше порядка последнего вычисляемого знака. Именно поэтому и не удается полностью реализовать возможности памяти МК-85м и ограничиться вычислением 4093 цифр, так как при вычислении следующей происходит выход за пределы цифрового диапазона микрокомпьютера.

В табл. 3 показаны начальное и конечное значения управляющих переменных A и B, число дополнительно открываемых регистров памяти R, необходимых для записи N знаков числа e. В последнем столбце указано время T, потраченное на эти вычисления микрокомпьютером МК-85м № 000140 в режиме быстродействия. Для вычисления 4093 знаков у автора не хватило пороха и цифры в последней строке получены расчетным путем.

Таблица 3

Вычисление основания натуральных логарифмов e

N	A	B	R	T
20	22	1	-	25 с
50	42	4	-	1 мин 50 с
100	70	9	-	6 мин
500	254	49	30	1 ч 50 мин
1000	450	99	80	6 ч 30 мин
1500	633	139	130	13 ч 45 мин
3310	1244	330	311	59 ч 36 мин
4093	1493	409	390	около 90 ч

Для дословного перевода с языка микрокалькулятора на язык микрокомпьютера нужно создать подпрограммы, выполняющие те же функции, что и операторы микрокалькулятора. Будем считать, что они записаны в файле 9, хотя это абсолютно не обязательно. Важно соблюсти правила перевода, а будет ли это подпрограмма или фрагмент программы, не суть важно.

Условимся, что регистрам памяти калькулятора P0, P1...P9, PA...PE будут соответствовать регистры A(0), A(1)...A(14). Регистрам стека X, Y, Z, T и X1 соответственно X, Y, Z, V и W. Правила формального перевода приведены в табл. 4, а подпрограммы, используемые при этом, в табл. 5. Так как логические операции в микрокалькуляторах исполь-

зуются преимущественно для создания текстовых сообщений, то они из этого рассмотрения исключены, но создание программ для их перевода — задача вполне разрешимая.

Таблица 4

Правила перевода с языка МК-61 на версию Бейсика МК-85

МК-61	МК-85
ПN	A(N)=X
КПN	N=A(N): A(N)=X
В↑	COSUB# 9,10
ИПN	GOSUB# 9,10: X=A(N)
КИПN	GOSUB# 9,10: N=A(N): X=A(N)
π	GOSUB# 9,10: W=X : X=π
одноместная операция (ОМО)	GOSUB# 9,20: X=ОМО X
двухместная операция (ДМО)	GOSUB# 9,30: X = X ДМО
↔	GOSUB# 9, 40
О	GOSUB 9, 50
Вх	GOSUB# 9, 60

Метод создания стека, продемонстрированный на примере перевода, имеет более широкое применение. Стек может применяться, когда работают с различными уровнями при управлении памятью. Да и в нашем МК-85 работа со скобками, с подпрограммами и циклами, вложенными друг в друга, осуществляется с помощью стека.

Таблица 5

Подпрограммы, имитирующие работу стека

```
10 V = Z : Z = У : У = X : RETURN
20 W = X : RETURN
30 W = X : X = У : У = Z : Z = V :  RETURN
40 W = X : X = У : У = W : RETURN
50 W = X : X = У : У = Z : Z = V :  V= :W :  RETURN
60 V = Z : Z = У : У = X : X = W :  RETURN
```

В качестве упражнения переведем программу Л.Габрида "Числа Фибоначчи" (Наука и жизнь, № 10/90, с. 113).

Программа 15. *Числа Фибоначчи*

МК-61	МК-85
00.ХП0	1 INPUT"N=",A
01.1 02.Flg	10 X=1:GOSUB #9,20:X=LOG X
	20 FOR A TO 1 STEP -1
03.FBx	30 GOSUB #9,60
04. ↔	40 GOSUB #9,40
05.+	50 GOSUB #9,30:X=X+W
06.FL0 07.03	60 NEXT A
08.С/П	80 PRINT X
09.БП 10.03	90 GOTO 20

В программу вводят порядковый номер вычисляемого члена ряда Фибоначчи N и получают его значение. Если после этого осуществить пуск клавишей EXE, то получим сначала N+1-й член, потом N+2-й и т.д. Конечно, изящная программа получилась громоздкой, но когда неизвестен алгоритм и математическая основа, на которой он построен, такой путь во многих случаях оправдан.

ДОГОВАРИВАТЬ ВСЕ ДО КОНЦА — ВЕРНЫЙ СПОСОБ НАСКУЧИТЬ

Ошибки и их диагностика. Программист делает ошибки двух типов: синтаксические, когда нарушаются правила языка программирования, и логические, когда программа отклоняется от алгоритма. Ниже речь пойдет только об ошибках первого типа. Они бывают трех видов: не выявляющиеся диагностическими средствами, проявляющиеся только при работе программы и, наконец, обнаруживаемые при вводе программы (строки) в память.

К первому виду относятся часто встречающиеся ошибки интерпретации формул. Например: *в*/2*ас*=B/2*A*C. Здесь знаменатель не взят в скобки, в результате будет вычислено выражение *abc*/2.

А вот список ошибок второго вида, за полноту которого автор не ручается:

пропущены разделители ",", ";" и ":"

нет знаков математических действий

пропущены переменные

отсутствуют аргументы у функций и ряда операторов

Для примера рассмотрим строку:

```
10 A$="текст":C=-B/(2*A) :P=(K+4)/5: PRINT B,C,"конец"
```

Будем считать, что у программиста, вводящего эту строку в память МК-85, произошло ужасное умопомрачение и он ввел ее так:

```
10=текст"/(2*:K4)/ BC "конец"
```

После нажатия клавиши EXE стока запишется в память как ни в чем ни бывало, а ошибки начнут диагностироваться только при отработке программы. Следует обратить внимание и на то, что при диагностике подсчитывается число кавычек и скобок. Сообщение об ошибке выводится только, если число кавычек нечетное или если число открытых и закрытых скобок не совпадает. Двойная ошибка,наподобие той, что произошла в строке 10, при вводе программы не диагностируется и выявляется только при прогоне.

При работе с ошибками следует иметь в виду, что курсор не всегда точно указывает на место ошибки. Например, когда нет левой скобки, то он стоит под правой, а если нет правой — в конце строки. Там же он будет при нечетном числе кавычек. При ошибке в аргументе функции или оператора курсор может либо точно указывать место ошибки, либо стоять под первой буквой названия оператора.

Во многих версиях Бейсика аргумент оператора функции должен обязательно заключаться в скобки. В МК-85 это не обязательно. Например запись ACS SGN ATN X почти во всех версиях Бейсика признается ошибочной. Там нужно писать ACS (SGN (TAN (X))). Наш компьютер признает обе формы записи правильными и ведет вычисления справа налево. Сначала вычисляется арктангенс X, потом определяется знак результата (-1, 0, 1) и далее находится арккосинус этого результата. Несмотря на то что отсутствие скобок в приведенном примере делает программу на шесть шагов короче, многие программисты пренебрегают этой экономией и заключают аргумент в скобки. Примеры можно встретить в приведенных далее программах.

На этом закончим первое знакомство с микрокомпьютером "Электроника МК-85" и перейдем непосредственно к программам.

БИБЛИОТЕКА ПРОГРАММ

МАТЕМАТИКА ЦЕЛЫХ ЧИСЕЛ

Помещенные в этом разделе программы объединяет одно — они оперируют с целыми числами.

Программа 16. *Факториал*

```
10 INPUT"N",A:B=1: IF A=0;A=1
20 FOR A TO 1 STEP -1
30 B=A*B:NEXT A
40 PRINT B:GOTO 10
```

В этой короткой программе две особенности.

Во-первых, вычисление факториала ведется в порядке убывания чисел, а во-вторых, присвоение значения управляющей переменной цикла А произведено до начала цикла, что позволяет использовать ее в строке 20 без знака присвоения.

Во многих версиях Бейсика есть функция A mod B — определение остатка от деления А на В. В нашей версии его нет, и для определения приходится пользоваться специальной программой или вводить в программу соответствующий фрагмент, как было сделано в программе 19.

Программа 17. *Определение остатка от деления А на В*

```
5 ! A mod B
10 PRINT "A mod B"
20 INPUT "A=",A,"B=",B
30 C=ABS A-ABS(B*INT(A/B))
40 PRINT C: GOTO 20
```

Почти во всех книгах по началам программирования приведены программы определения наибольшего общего делителя числа А и В. Не будем и мы отступать от этой традиции.

Программа 18. *НОД А и В*

```
1( PRINT "НОД A и B"! A и B - целые
20 INPUT "A=",A,"B=",B:
    IF A = 0 THEN 80
30 IF B=0 THEN 80
40 X=A:Y=ABS B:IF A<B;Y=ABS A:X=B
50 R=X-Y*INT(X/Y)
60 IF R=0;PRINT "НОД A,B=";Y:GOTO 10
70 X=Y:Y=R:GOTO 50
80 PRINT "Повторить ввод.A или B=0":
    GOTO 20
```

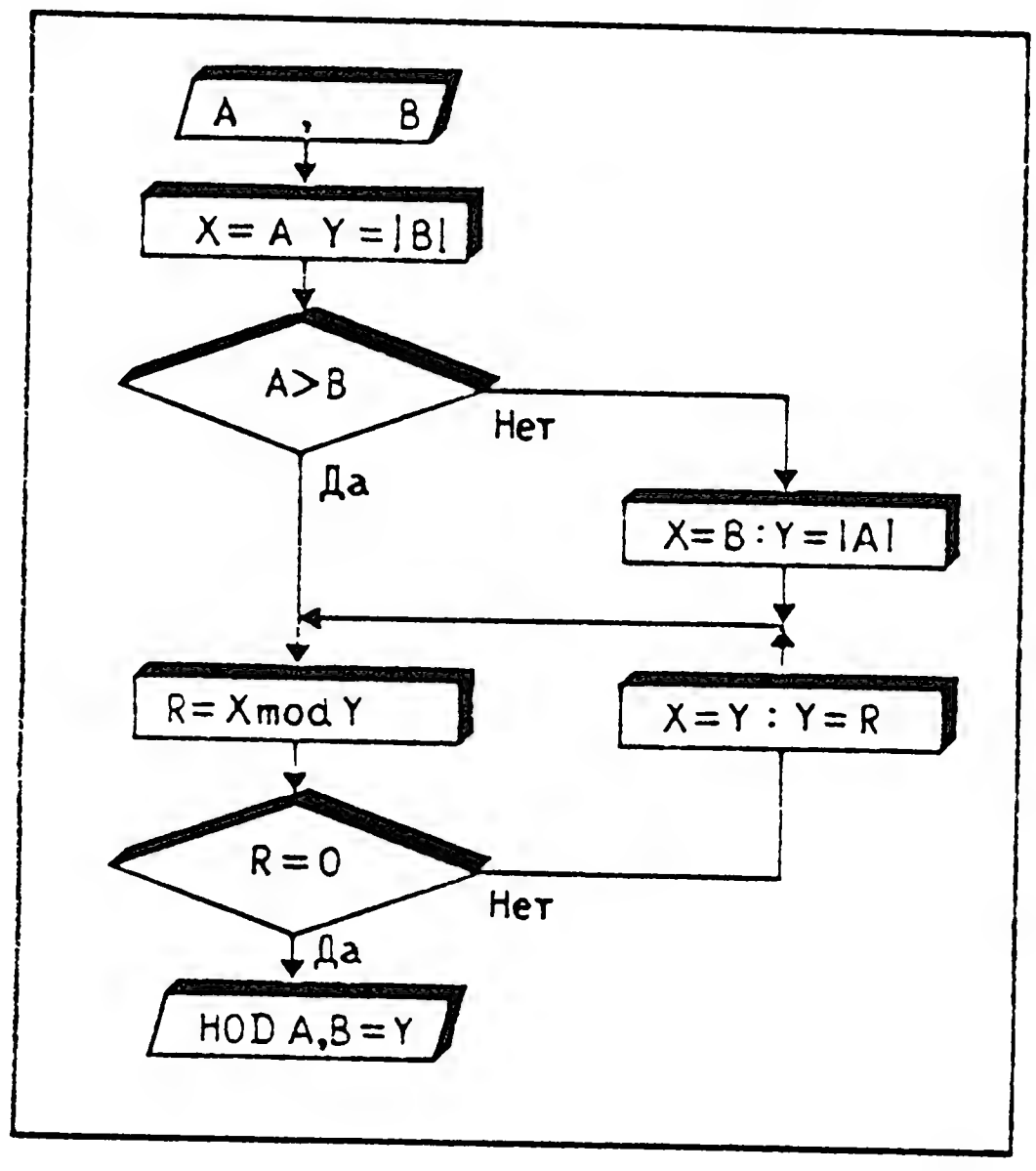


Рис.5. НОД А, В (алгоритм ЕВКЛИДА)

При составлении программы применен алгоритм Евклида, показанный на рис. 5. В этом алгоритме используются следующие свойства НОД:

НОД(2m,2n)=2НОД(m,n)

НОД(2m,2n+1)=НОД(m,2n+1)

НОД(-m,n)=НОД(m,n)

При работе программы число X делится на такую степень числа 2, чтобы в результате получилось нечетное число. Если имеем X=У, то НОД А,В=У определен, в

противном случае переменным присваивают значения X=У и У=Р, и процесс повторяется.

Когда известен НОД, то легко определяется и наименьшее общее кратное — НОК(A,B)=A*B/НОД(A,B)

Программа 19. *НОК A и B*

```
10 !НОК A и B
20 INPUT "A=",A,"B=",B
IF A=0 THEN 80
30 IF B=0 THEN 80
40 X=A:Y=ABS B:IF A<B;
Y=ABS A:Y=B
50 R=X-Y*INT(X/Y)
60 IF R=0 THEN 90
70 X=Y:Y=R:GOTO 50
80 PRINT"повторить ввод
A или B=0":GOTO 20
90 X=A*B/Y:PRINT X:
GOTO 20
```

В этой программе сначала вычисляется НОД, а потом по приведенной выше формуле — НОК.

Считают, что одним из первых свойств чисел, открытых человеком, — это возможность некоторых из них разлагаться на множители. Это делает программа 20.

Программа 20. *Разложение на множители*

```
10 INPUT "Разложить "A:
   IF A=0 THEN 30
20 IF A=INT(A) THEN 40
30 PRINT "Множ. нет": GOTO 10
40 FORJ=2 TO A
50 K=0
60 IF A/J <> INT(A/J) THEN 80
70 A=A/J:K=K+1: GOTO 60
80 IF K=0 THEN 100
90 PRINT J; "^"; K
100 IF A=1 ; PRINT"разложе-
    ние закончено":GOTO 10
110 NEXT J
```

Разложение осуществляется с помощью цикла, а результат выводится в формате "J ↑K-", где J — очередной множитель, K показывает, сколько раз он повторяется. При каждом нажатии на клавишу EXE на индикатор выводится следующий множитель. Когда все они определены, то появляется сообщение "разложение окончено". Программа защищена от ввода нуля и чисел, имеющих дробную часть. При их вводе выдается сообщение "множителей нет".

Сведущие люди могут сыграть злую шутку, предложив кому-нибудь разложить на множители число 4 294 967 297. В режиме быстродействия, проработав минуту с небольшим, компьютер выдает результат "641 ↑1", но второго результата придется ждать долго. Дело в том, что число, предложенное для разложения, это пятое число Ферма, являющееся произведением двух простых чисел: F_5=4 294 967 297=641*6 700 417. Очередной испытуемый множитель формируется нашим компьютером в цикле от 2 до N, и если N простое, то цикл повторяется N — 2 раза, а поэтому на поиск второго множителя пятого числа Ферма уйдет времени в 10453 раза больше, чем на поиск первого. У кого хватит терпения ждать тот может дождаться результата.

УНИВЕРСАЛЬНАЯ ПРОГРАММА ДЛЯ РЕШЕНИЯ ТРЕУГОЛЬНИКОВ

Из всех геометрических задач решение треугольников встречается чаще всего. По заданным трем элементам нужно определить остальные. В число заданных элементов должна входить по крайней мере одна сторона. Кроме определения неизвестных сторон и углов, иногда требуется определять площадь, радиусы вписанной и описанной окружностей, биссектрисы, медианы или высоты.

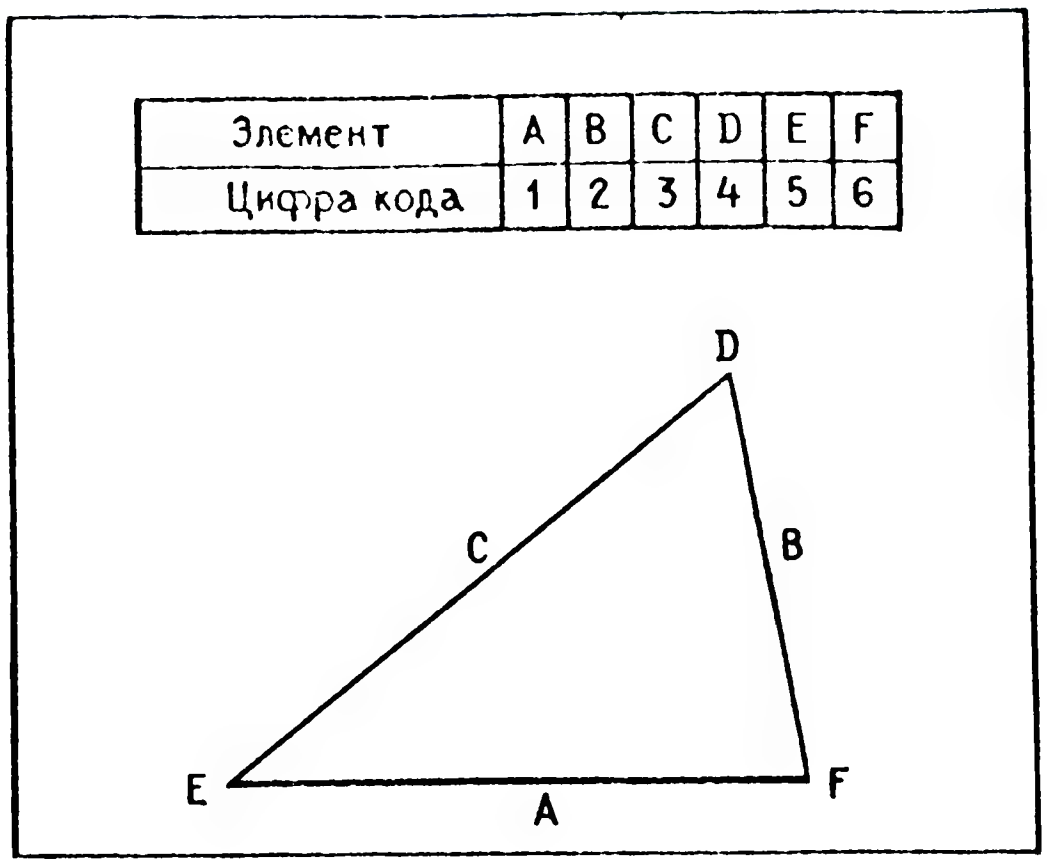


Рис.6. Решение треугольников

На рис. 6 показаны элементы треугольника и цифры, которыми они кодируются. В МК-85 имя переменной может быть только прописной буквой латинского алфавита, поэтому стороны и углы обозначены немного необычно. Конечно, можно было бы прибегнуть к обычным обозначениям, но тогда нужно было бы интерпретировать символы в программе. Сейчас этого делать не надо, так как они совпадают с обозначениями на рисунке. Для определенности будем считать, что сторона A всегда задана, код варианта при этом можно сократить до двух цифр, обозначающих остальные известные элементы треугольника. Возможны следующие варианты, отраженные в следующем плане разработки программы.

Даны три стороны A, B и C (код 23)
Решение: D=(B*B*C*C-A*A)/(2*B*C)
IF ABS(D) >1 THEN нет решения (треугольник невозможен)
D=ACS(D): E=ASN (B*SIN D/A):F=ACS (-1)-D-E

Функция ACS (-1) использована для того, чтобы получить сумму углов треугольника в том формате, который задан клавишей установки режима (MODE 4 — градусы, MODE 5 — радианы и MODE 6 -грады). Прием полезно запомнить, так как он встречается во многих программах, где используются тригонометрические функции. Ранее он уже использовался в программе 9.

Даны две стороны и угол, лежащий против одной из них, A,B и D (код 24).
Решение: IF A<B AND B*SIN D>A THEN нет решения.
IF A<B AND B*SIN D=A THEN прямоугольный треугольник (угол E — прямой).
IF A< B AND B*SIN D<A THEN два решения.
В первом E1=ASN(B*SIN D/A), а во втором — E2=ACS(-1)-E1
F=ACS(-1)-D-E : C=A*SIN F/SIN D.

Даны две стороны и угол, лежащий между ними A, B и F (код 26).
Решение: C=SGR (A*A+B*B-2*A*B*COS F) : D=ASN(A*SIN F/C)
IF B/A<COS F THEN D=ACS(-1)-D: E=ACS(-1)-D-E

Даны сторона и два прилежащих к ней угла A, E и F (код 56)
Решение: D=ACS(-1)-E-F. Стороны определяются по теореме синусов, как и в предыдущем случае.

Даны стороны, прилежащий и противолежащий угол A, D и F (код 45).
Решение: E=ACS(-1‘-D-F). Остальное как в предыдущем варианте.

Если заданы другие элементы, то путем циклической перестановки или взятия зеркального отображения все они сводятся к рассмотренным вариантам, в которых площадь определяется как S=B*C*SIN D/2, радиус описанной окружности — R=A*B*C/(4*S) и радиус вписанной окружности r=2*S/(A+B+C). Этот план реализован в приведенной ниже программе.

Программа 21. *Решение треугольников*

```
5 !ТРЕУГОЛЬНИК
10 INPUT "код",G
15 IF G<>23;IF G<>24;IF G<>26;
IF G<>45;IF G<>56 THEN 10
20 GOTO G
23 GOTO 240
24 GOTO 100
26 GOTO 190
45 GOTO 80
56 INPUT"a=",A,"B=",E,"C=",F:
D=ACS(-1)-E-F
60 B=A*SIN E/SIN D
70 C=A*SIN F/SIN D:GOSUB 300:
GOTO 10
80 INPUT "a=",A,"A=",D,"B=",E
90 F=ACS(-1)-D-E:GOTO 60
```

```
100 INPUT"a=",A,"b=",B,"A=",D
110 IF A>=B THEN 180
120 IF B*SIN D>A THEN 270
130 IF B*SIN D=A;E=ACS(-1)-A-D:
GOTO 70
140 PRINT"ДВА РЕШЕНИЯ.ПЕРВОЕ_";:
E=ASN(B*SIN D/A)
150 F=ACS(-1)-D-E:C=A*SIN F/SIN D:
GOSUB 300
160 PRINT"ВТОРОЕ РЕШЕНИЕ_";:
170 E=ACS(-1)-E:F=ACS(-1)-D-E:
GOTO 70
180 E=ASN(B*SIN D/A):
F=ACS(-1)-D-E:GOTO 70
190 INPUT"a=",A,"b=",B,"C=",F
200 C=SQR(A*A+B*B-2*A*B*COS F)
10 D=ASN(A*SIN F/C)
220 IF B/A<COS F;D=ACS(-1)-D
230 E=ACS(-1)-D-F:GOTO 70
240 INPUT"a=",A,"b=",B,"c=",C
250 D=(B*B+C*C-A*A)/(2*B*C)
260 IF ABS D<=1 THEN 280
270 PRINT"НЕТ РЕШЕНИЯ":GOTO 10
280 D=ACS D:E=ASN(B*SIN D/A)
290 F=ACS(-1)-D-E:GOSUB 300:
GOTO 10
300 S=B*C*SIN D/2:
R=A*B*C/(4*S):T=2*S/(A+B+C)
310 PRINT"a=";A,"b=";B,"c=";C,
"A=";D,"B=";E,"C=";F
320 PRINT"S=";S,"R=";R,"r=";T:
RETURN
```

После пуска программы на индикаторе запрос кода в формате "код?". Если введен код, отсутствующий в списке, запрос повторяется. Код одновременно является и адресом решаемого варианта. В формате вывода элемента присутствует его символьное обозначение. Каждый последующий элемент выводится после нажатия клавиши EXE. После того как все они выведены, управление передается на начало программы, и можно решать следующий треугольник. Если это необходимо, то можно ввести проверку на знак, и в случае введения отрицательных чисел предлагать повторить ввод. В этой программе такая защита не предусмотрена.

РЕГРЕССИВНЫЙ АНАЛИЗ И КОЭФФИЦИЕНТ КОРРЕЛЯЦИИ

Трудно переоценить роль вычислений, ведь они часто бывают единственной возможностью добраться до истины, особенно когда речь идет об обработке данных, полученных экспериментально. Из всего множества задач, связанных с такой обработкой, остановимся на двух. На аппроксимации массива функцией и сглаживании экспериментальной зависимости наиболее подходящей функций. Огрничимся исследованием непериодических экспериментальных кривых, не имеющих разрывов и точек перегиба. Будем полагать, что отклонение точек x и y от истинных значений невелико, что позволит использовать метод наименьших квадратов. Нужно учесть и то, что найденная функция не обязательно будет математической моделью исследуемого явления. Просто из того набора, который мы выберем, она будет наилучшим образом описывать нашу экспериментальную кривую на заданном интервале.

В основу метода поиска аппроксимирующей функции $y=F(x)$ положен метод линеаризации, т.е. подбора функций $Y=\psi(y)$ и $X=\chi(x)$, которые позволят заменить исследуемую функцию линейной $Y=AX+B$.

Достаточно удачный набор аппроксимирующих функций предложил Г.Славин (Наука и жизнь, №2/88, с.132). С незначительными изменениями он приведен в табл. 6. Там же указывается, какое значение присваивается переменным X и Y при линеаризации функции. Подставляя эти значения в уравнение прямой

$$\frac{X_3-X_1}{X_2-X_1}-\frac{Y_3-Y_1}{Y_2-Y_1}=F(x,y)$$

ищут методом перебора минимальную величину F(x,y). Функция, соответствующая этому минимуму и будет наилучшим приближением из нашего набора,а если она еще и близка к нулю, то данная функция обеспечивает наилучшую аппроксимацию. Про-

грамма 22 производит перебор функций и ищет минимум F(x,y). По запросу компьютера оператор вводит три пары точек, после чего начинается поиск подходящей функции. После завершения счета ее код выводится на индикатор. Для того чтобы узнать значение F(x,y), нужно нажать на клавишу EXE. Следующее нажатие передаст управление на начало программы, и компьютер готов к вводу новых трех пар точек.

Программа 22. *Подбор аппроксимирующей функции*

```
10 INPUT"X1=",A,"X2=",B,"X3=",C
20 D=A:E=B:F=C:GOSUB 200:G(1)=G
30 D=A*A:E=B*B:F=C*C:GOSUB 200:G(2)=G
40 D=1/A:E=1/B:F=1/C:GOSUB 200:G(3)=G
50 D=LN A:E=LN B:F=LN C:GOSUB 200:
G(4)=G
60 INPUT"Y1=",A,"Y2=",B,"Y3=",C:
M=1E9:N=0
70 D=A:E=B:F=C:GOSUB 200:GOSUB 210
80 D=A*A:E=B*B:F=C*C:GOSUB 200:GOSUB 210
90 D=1/A:E=1/B:F=1/F:GOSUB 200:
GOSUB 210
100 D=LN A:E=LN B:F=LN C:
GOSUB 200:GOSUB 210
110 PRINT P,M,"конец":GOTO 10
200 G=(F-D)/(E-D):RETURN
210  FOR L=1 TO 4
220 N=N+1:Q=ABS(G-G(L))
230 IF Q<M;M=Q:P=N
240 NEXT L:RETURN
```

Эта программа ведет выбор из шестнадцати функций табл. 6 (коды 1 — 16).

Если произвести замену натурального логарифма на десятичный, то в сравнении будут участвовать функции, помещенные в знаменателе строк 4, 8, 12, 13 — 16 табл. 6.

Следующая программа использует метод наименьших квадратов для нахождения коэффициентов a и b в исходной функции, являющейся функцией наибольшего приближения. Таким образом, происходит сглаживание исходных данных. При работе со степенными функциями некоторые коэффициенты нужно пересчитать. Как это сделать, ясно из той же табл. 6, в которой два последних столбца дают соотношение между *a* и *A*, *b* и *B*. В эту же таблицу введены функции, не определяемые предыдущей программой (коды 17 — 22). Если известно, что именно среди них находится функция наилучшего приближения, то, введя ее код, получим коэффициенты этой функции *A* и *B*. Ввод пар происходит по запросу компьютера. Кроме коэффициентов *A* и *B*, рассчитывается и коэффициент корреляции *r*. Чем он ближе к 1, тем надежнее полученные результаты.

Программа 23. *Линейное сглаживание прямой У=АХ+В*

```
5 !regressio-3
10 VAC:INPUT"код",Z,"N=",N:Z=Z+80
20 FOR W=1 TO N
30 PRINT"X";W;"=";:INPUT X:PRINT
"Y";W;"=";:INPUT Y:GOSUB Z
40 C=C+X:D=D+Y:E=E+X*X:F=F+Y*Y:
G=G+X*Y:NEXT W
50 A=(G-C*D/N)/(E-C*C/N)
60 B=(D-A*C)/N
70 R=(G-C*D/N)/SQR((E-C*C/N)*
(F-D*D/N))
80 PRINT"A=";A,"B=";B,"r=",ABS R
81 RETURN
82 X=X*X:RETURN
83 X=1/X:RETURN
84 X=LN X:RETURN
85 Y=Y*Y:RETURN
86 X=X*X:Y=Y*Y:RETURN
87 X=1/X;y=y*y:RETURN
88 X=LN X:Y=Y*Y:RETURN
89 Y=1/Y:RETURN
90 X=X*X:Y=1/Y:RETURN
91 X=1/X:Y=1/Y:RETURN
92 X=LN X:Y=1/Y:RETURN
93 Y=LN Y:RETURN
94 X=X*X:Y=LN Y:RETURN
95 X=1/X:Y=LN Y:RETURN
96 X=LN X:Y=LN Y:RETURN
97 Y=LOG Y:RETURN
98 X=X*X:Y=LOG Y:RETURN
99 X=1/X:Y=LOG Y:RETURN
100 X=LOG X:Y=LOG Y:RETURN
101 Y=Y/X:RETURN
102 X=EXP X:RETURN
```

Таблица 6

Преобразование функции y=x в линейную функцию У=АХ+В

код	функция y= F(x)	X	У	a	b
1	$y=ax+b$	x	y	A	B
2	$y=ax^2+b$	x^2	y	A	B
3	$y=a/x+b$	1/x	y	A	B
4	$\frac{y=a\ln x+b}{y=a\lg x+b}$	$\frac{\ln x}{\lg x}$	$\frac{y}{y}$	$\frac{A}{A}$	$\frac{B}{B}$
5	$\sqrt{ax+b}$	x	y^2	A	B
6	$\sqrt{ax^2+b}$	x^2	y^2	A	B
7	$D\sqrt{a/x+b}$	1/x	y^2	A	B
8	$\frac{y=\sqrt{a\ln x+b}}{y=\sqrt{a\lg x+b}}$	$\frac{\ln x}{\lg x}$	$\frac{y^2}{y^2}$	$\frac{A}{A}$	$\frac{B}{B}$
9	$y=1/(ax+b)$	x	1/y	A	B
10	$y=1/(ax^2+b)$	x^2	1/y	A	B
11	$y=1/(a/x+b)$	1/x	1/y	A	B
12	$\frac{y=1/(a\ln x+b)}{y=1/(a\lg x+b)}$	$\frac{\ln x}{\lg x}$	$\frac{1/y}{1/y}$	$\frac{A}{A}$	$\frac{B}{B}$
13	$\frac{y=be^{ax}}{y=ba^x}$	$\frac{x}{x}$	$\frac{\ln y}{\lg y}$	$\frac{A}{10^A}$	$\frac{e^B}{10^B}$
14	$\frac{y=be^{ax^2}}{y=ba^{x^2}}$	$\frac{x^2}{x^2}$	$\frac{\ln y}{\lg y}$	$\frac{A}{10^A}$	$\frac{e^B}{10^B}$
15	$\frac{y=be^{a/x}}{y=ba^{1/x}}$	$\frac{1/x}{1/x}$	$\frac{\ln y}{\lg y}$	$\frac{A}{10^A}$	$\frac{e^B}{10^B}$
16	$y=bx^a$.	$\frac{\ln x}{\lg x}$	$\frac{\ln y}{\lg y}$	$\frac{A}{A}$	$\frac{e^B}{10^B}$
17	$y=ba^x$	x	lgy	10^A	10^B
18	$y= ba^{x^2}$	x^2	lgy	10^A	10^B
19	$y= ba^{1/x}$	1/x	lgy	10^A	10^B
20	$y=bx^a$	lgx	lgy	A	10^B
21	$y=x/(ax+b)$	x	y/x	A	B
22	$y=ae^x+b$	EXP x	y	A	B

ДЕЛУ ВРЕМЯ — ПОТЕХЕ ЧАС

Во время повального увлечения оздоровительным бегом был очень популярен 12-минутный тест К.Купера, предназначенный для оценки степени физической подготовленности. Напомним, что этот тест применим либо к здоровым людям моложе 35 лет, либо к более старшим по возрасту, но выполнившим по крайней мере задания первых шести недель программы К.Купера. За 12 минут нужно пробежать по ровной местности как можно дальше. Если во время теста появятся болезненные симптомы, то бег прекращается.

Для оценки степени физической подготовленности К.Купером была разработана специальная таблица, которую воспроизводит программа 24. При этом в компьютер заложена не сама таблица, а математический алгоритм, по которому она вычислялась.

Программа 24. *Бег — тест К.Купера*

```
10 PRINT "пол М-1;Ж-2";:INPUT A
20 INPUT "имя",$,"лет",B,"D",D
30 FOR E=0 TO 6
40 B=B-A*10
50 IF B<0 THEN 70
60 NEXT E
70 D=D+.15*E*A
80 IF D>2.7;PRINT"ОТЛИЧНО";$:GOTO 10
90 IF D>2.5;PRINT"ХОРОШО";$:GOTO 10
100 IF D>2.2;PRINT"НЕ ПЛОХО";$:GOTO 10
110 IF D>2.1;PRINT"ПЛОХО";$;GOTO 10
120 PRINT"Оч.ПЛОХО";$:GOTO 10
```

По запросу компьютера вводится пол (1 — мужской, 2 — женский), имя, возраст и дистанция, пройденная за 12 минут. После каждого ввода нажимается клавиша EXE. После последнего ввода программа выдает степень физической подготовленности, после чего управление передается на начало программы.

Пользуясь аналогичным приемом, можно составить программы для тестов Купера — ходьба, плавание и велосипед

Из-за своей простоты большой популярностью пользуется тест "Ступени", применяемый для оценки физического состояния. В течение пяти минут нужно подниматься на ступеньку высотой 25 — 30 см и спускаться с нее, подсчитывая число циклов "подъем/спуск" (N). После окончания упражнения сосчитать удары пульса: с 60 до 90 секунд (P1), со 120 по 150 секунд (P2) и с 180 по 210 секунд (P3). По запросу компьютера, в него вводится имя тестируемого, N, P1, P2 и P3. Еще раз напомним и больше этого делать не будем, что после каждого ввода нажимается клавиша EXE. После последнего ввода программа выдает оценку физического состояния по пятибалльной шкале.

Программа 25. *Ступени*

```
5 !Harvard step-test
10 INPUT "имя",$,"N",A,"P1",B,"P2",C,"P3",D
20 E=10*A/(B+C+D)
30 IF E>9;PRINT "ОТЛИЧНО";$:GOTO 10
40 IF E>7;PRINT "ХОРОШО";$:GOTO 10
50 IF E>6;PRINT "не плохо";$:GOTO 10
60 IF E>5;PRINT "ПЛОХО";$:GOTO 10
70 PRINT "Оч.плохо";$:GOTO 10
```

Программа "Восточный гороскоп" использует то обстоятельство, что в 12-летних циклах восточных календарей у каждого года есть свой покровитель из фауны. Между покровителями могут существовать дружеские отношения, нейтральные или они могут быть врагами. Считается, что и у людей, родившихся в соответствующие годы, отношения с другими людьми складываются так же, как и у покровителей этих лет.

Программа 26. *Восточный гороскоп*

```
10 A$="-МЫШЬ":B$="-БЫК":
C$="-ТИГР":D$="-КОТ":
20 E$="-ДРАКОН":F$="-ЗМЕЯ":
G$="-КОНЬ":H$="-КОЗА"
30 I$="-МАКАКА":J$="-ПЕТУХ":
K$="-СОБАКА":L$="-КАБАН"
40 INPUT "ГОД РОЖД",Z
50 Z=Z-1924-12*INT((Z-1924)/12):
IF Z<0;Z=Z+12
60 PRINT "ВЫ";A$(Z):X=Z
70 PRINT "ВРАГИ";
80 FOR Y=1 TO 3
90 X=X+3:IF X>=12;X=X-12
100 PRINT A$(X);:NEXT Y:STOP
110 X=Z:PRINT "ДРУЗЬЯ";
120 FOR Y=1 TO 2
130 X=X+4:IF X>=12;X=X-12
140 PRINT A$(X);:NEXT Y:STOP: GOTO 40
```

В программу вводят свой год рождения или какой-то другой, для которого нужно получить необходимые сведения, и, нажимая клавишу EXE, последовательно получают имя покровителя введенного года, имена врагов — их три, и друзей — их два. После этого управление передается на начало программы.

Игра американских школьников "Быки и коровы" стала популярной и у нас благодаря публикации ряда ее компьютерных версий. Компьютер загадывает число, а игрок, вводя пробные числа, пытается его отгадать. Компьютер сообщает, сколько в пробном числе есть "коров" (цифр, совпадающих с цифрами задуманного числа) и "быков" (угаданных цифр, стоящих на том же месте, что и в задуманном числе). Особенностью публикуемой программы М.Вулиса, доработанной автором, является то, что компьютер может загадывать число с повторяющимися цифрами, что делает игру более сложной. После пуска программы на индикаторе четыре вопросительных знака, говорят о том, что компьютер задумал четырехзначное число и просит ввести пробное. После ввода пробного числа и нажатия клавиши EXE оно повторяется на индикаторе слева, на этой же строке показано число "быков" и "коров", а вопросительный знак в конце строки предлагает ввести следующее пробное число. Когда число угадано, компьютер сообщает, за сколько попыток это сделано. Этот вариант игры назван "Логика", так как логические умозаключения играют не последнюю роль при отгадывании числа.

Программа 27. *Логика*

```
1 PRINT "Logik"
5 VAC: A = RAN# * 10000:
IF A < 1000 THEN 5: GOSUB 70
10 D=0:INPUT"???",$:GOTO 30
20 CSR 10:INPUT $
30 A=VAL $:B=0:C=0:D=D+1:E=4:
GOSUB 70: FOR E = 1 TO 4: A = E(E)
40 IF A=I(E);B=B+1:IF B=4;PRINT
" за ";D;" раз":GOTO 5
50IF(A-J)*(A-K)*(A-L)*(A-M)=0;C=C+1
60 NEXT E:PRINT $;B;C;:GOTO 20
70 FOR E TO E+3:A=INT A/10 :
F(E)=FRAC A:
NEXT E: RETURN
```

Большая группа игр сводится к вынимании из урны шаров с номерами. В тираже могут участвовать все N шаров, находящихся в урне и вынимаемых случайным образом или только n из них ($n \leq N$). Следующие две программы моделируют лототрон и с ними можно проводить различные тиражи, в которых будет участвовать n шаров из N, нахдящихся в урне ($n \leq N$). Различие между этими программами в том, что в "Лототроне-1" в памяти находятся номера всех N шаров.

В играх типа лото, память представляет собою мешок, в котором находятся все номера. ГСЧ случайным образом вызывает на индикатор содержимое того или иного регистра, а чтобы он больше не участвовал в тиражах, его содержимому приписывается отрицательное значение. После того как n номеров выпало, появляется сообщение "тираж закончен", но нажатие на клавишу EXE приведет к появлению на индикаторе выигравших номеров в порядке их возрастания.

В "Лототроне-2" ГСЧ выдает числа из требуемого диапазона, и при генерации последующего числа происходит проверка, нет ли его уже среди выпавших. Если есть, то число бракуется и ГСЧ дает новое, которое так же проверяется. Такой метод требует значительно меньше регистров памяти и применяется когда проектируются игры типа "Спортлото 5 из 36", так как хранить в памяти нужно только выигравшие числа.

Работают с программами одинаково. После запуска по запросу вводят число шаров в урне N и число вынимаемых шаров n. После этого каждое нажатие на клавишу EXE приводит к появлению на индикаторе выигравшего (выпавшего) номера. Если программа 29 используется для многократного проведения тиражей одной и той же лотереи, то целесообразно исключить из строки 80 оператор DEFM 0 и изменить адрес безусловного перехода, сделав его GOTO 30. Это избавит от ввода N и n перед каждым новым тиражом. Эти величины будут вводиться только перед самым первым пуском. А после окончания тиража нажатие клавиши EXE приведет сразу к началу следующего.

Программа 28. *Лототрон — 1*

```
5 ! лототрон-1
10 INPUT "N=",A:B=A
20 IF A>20;DEFM(A-20)
30 FOR A TO 1 STEP-1
40 E(A)=A:NEXT A
50 INPUT "n=",C
60 IF C>B;PRINT"еще раз_";:GOTO 50
70 D=INT(B*RAN#+1)
80 IF E(D)<0 THEN 70
90 PRINT E(D):E(D)=-E(D):
C=C-1
100 IF C<>0 THEN 70
110 PRINT"тираж окончен"
!NEW-Sn
120 FOR A=1 TO B
130 IF E(A)<0;PRINT -E(A)
140 NEXT A:DEFM 0:PRINT"конец":
GOTO 10
```

Программа 29. *Лототрон — 2*

```
5 ! лототрон-2
10 VAC:INPUT "N=",A,"n=",B
15 IF B>A;PRINT "еще раз_";:
GOTO 10
20 IF B>20 ;DEFM (B-20)
30 FOR C=1 TO B
40 D=INT(A*RAN#+1)
60 IF F(E)=D THEN 40
70 NEXT E:F(C)=D:PRINT D:
NEXT C
80 DEFM 0:PRINT"тираж закончен":
GOTO 10
```

Хорошее владение клавиатурой — залог быстрого и безошибочного ввода программы. Судя по присланным в редакцию письмам, владельцы МК-85 придумали много тренажеров, не только помогающих освоить клавиатуру, но и являющихся весьма интересными играми. Для публикации отобрана программа М.Кирсанова из Воронежа, в которую внесены небольшие изменения.

Программа 30. *Тренажер*

```
5 INPUT"N=",N,"M=",M
10 K=0:FOR I=1 TO N
:B$=CHR(65+26*RAN#):
PRINT B$;
20 FOR J=1 TO M:NEXT J:A$=KEY:
IF A$=B$;K=K+1
30 NEXT I:CSR 0:IF K<>N; PRINT"увы,всего ";K:STOP
40 PRINT"ЗНАТОК!"
```

После пуска программы компьютер запрашивает число букв в серии (N), и время нахождения буквы на индикаторе (M), за которое нужно успеть нажать соответствующую клавишу. Чем больше M тем это время больше.

Начинать тренировки рекомендуется с M=100. Буквы начинают появляться с нулевой позиции индикатора. Их коды врабатываются с помощью ГСЧ (строка 10). При появлении буквы нужно нажать соответствующую ей клавишу и не отпускать ее до появления следующей буквы. После того как вся серия из N букв показана, компьютер выдает число правильно нажатых клавиш, а если все они были нажаты правильно, то на индикаторе появляется надпись: "ЗНАТОК !". Возврат к началу программы осуществляется командой Sn, где n — номер файла, в котором записана программа. Когда не изменяются N и M, чтобы не вводить их каждый раз, последующие запуски осуществляют командой RUN 10.

Если в предыдущей серии все клавиши были нажаты правильно, то время индикации буквы уменьшается, а если нет, то увеличивается.

Соревнования по многим видам спорта судят N судей (гимнастика, фигурное катание и т.д.). При определении балла, как правило, отбрасываются самая высокая и самая низкая оценки, а остальные усредняются. Этот подсчет можно поручить МК-85.

TUTTI FRUTTI

К.Вяземский (г.Москва)

ГРАФИКИ ФУНКЦИЙ, РЕДАКТОР СПРАЙТОВ

В "Руководстве по эксплуатации МК-85" приведена программа, строящая график синусоиды. Для того чтобы МК-85 строил графики любых функций, достаточно повернуть систему координат на 90°. При этом ось ординат должна быть направлена вдоль индикатора. Теперь можно считать, что вдоль графика функции скользит узкая щель и снимаются ограничения по длине графика вдоль оси абсцисс. Эта идея реализована в программе "График функции".

Программа — график функции

```
1 MODE 5:INPUT"Хнач",X
2 INPUT"сдвиг по Y",B:B=B+30
3 INPUT"шаг по X",C,"шаг по Y",D
4 P=INT LOG C-1:GOSUB 20
5 CSR 0,:L=0:FOR I=1 TO 7:DRAW B,7-I
6 IF RND(X,P)=0;L=I
8 FOR J=1 TO E
9 Y=f(X)
10 Y=INT(B+Y/D):X=X+K:DRAW Y,7-I:
   NEXT J
11 NEXT I:IF L<>0;FOR T=1 TO 58:
DRAW T,7-L:NEXT T
12 F=ASCI KEY:IF F>=4;IF<=9 THEN 10+F
13 GOTO 12
14 X=X-7*C:GOTO 5
15 X=X-12*C:GOTO 5
16 X=X-2*C:GOTO 5
17 CSR 0,:PRINT RND(X,P);:GOTO 12
18 GOSUB 20:GOTO 12
19 CSR 0,:END
20 INPUT"точность",E:K=C/E:
   X=X+K/100:RETURN
```

Прежде всего необходимо в строку 9 вписать формулу функции, график которой будет строиться. После этого можно запускать программу. Первый вопрос, о начальном значении X, в комментариях не нуждается. "Сдвиг по У" — это смещение оси абсцисс вверх или вниз от середины индикатора. Например: когда строится график функции y=1/X, то при X>0 все значения функции положительны и есть смысл сдвинуть ось абсцисс в самый низ. Шаг по X — пределы изменения X, внутри которых значения функции будут лежать на одной вертикали. Так, при шаге 0,1 все точки с абсциссами $0 \le x < 0{,}1$ будут на одной вертикали, а $0.1 < X < 0.2$ — на соседней и т.д. Шаг по У — то же самое, но для оси ординат. Параметр "точность" определяет, сколько значений функции будет рассчитано внутри шага по X. При более высокой точности в крутых графиках на одной вертикальной линии будет больше точек, а это улучшает качество изображения. В то же время для пологих кривых требования к степени точности можно ослабить, что приведет к уменьшению числа рассчитываемых точек и более быстрому построению графика.

Управление программой осуществляется следующими клавишами:

→ сдвиг вправо, ← сдвиг влево (сдвиг происходит на 5 шагов, а два шага от предыдущего кадра всегда остаются).

EXE — построение участка графика без сдвига.

AC — изменение точности. Это бывает необходимо сделать при изменении крутизны графика.

DEL — вывод численного значения аргумента, соответствующего последней вертикальной линии.

ANS — конец работы.

Все сервисные функции вызываются только после заполнения кадра целиком. Чтобы программа безболезненно проходила точки разрыва II рода (X=0 для У=1/X и т.п.), к на-

чальному значению X прибавляется очень маленькая величина, не влияющая в пределах индикатора на точность построения графика. Наиболее эффективно программа работает в режиме быстродействия.

Следующая программа — редактор спрайтов — значительно облегчает задачу создания символа с кодом 96. Этот редактор с успехом применялся при разработке программы 4 "Профессиональный отбор" и в ряде других случаев.

Программа — редактор спрайтов

```
1 VAC:V=1:W=2:X=4:Y=8:Z=16
2 GOSUB 6:IF K=0;DRAWC 4-I,J
3 I=I+M:J=J+N:DRAW 4-I,J:M=0:N=0
4 Q=ASCI KEY:IF Q>=49;IF Q<=56
  THEN Q*2
5 GOTO 4
6K=INT(FRAC(A(J)/V(I)/2)*2):
  RETURN
98GOSUB 6:IF K=1;A(J)=A(J)-V(I)
99 GOTO 4
100 IF J>0;N=-1
101 GOTO 2
102 FOR I=0 TO 6:K=48+A(6-I):IF
      K>57;K=K+7
103 $=$+CHR K:NEXT I:PRINT $;:END
104 IF I<4;M=1
105 GOTO 2
106 GOSUB 6:IF K=0;A(J)=A(J)+V(I)
107 GOTO 4
108 IF I>0;M=-1
109 GOTO 2
110 GOTO 4
112 IF J<6;N=1
113 GOTO 2
```

Для управления работой программы используют цифровые клавиши: 4 — влево, 6 — вправо, 8 — вверх, 2 — вниз, 5 — установить точку, 1 — стереть точку и 3 — завершить работу. Результатом выполнения программы является последовательность символов, кодирующая вновь созданный спрайт.

Зверев Д. (г.Москва)

GETC ВМЕСТО DATA

В версии Бейсика для МК-85 нет оператора DATA, но его можно заменить оператором GETC при следующих условиях: в аргументе DATA оператор и его аргумент полностью размещаются в одной строке, в аргумент входят однозначные положительные числа. Нетрудно подсчитать, что если номер строки однозначный, то в аргументе может быть не более чем 36 чисел. На самом деле их несколько меньше, так как из текстовой постоянной еще нужно сделать число.

Вот программа, заменяющая оператор DATA.

Программа — оператор DATA

```
5 ! GETC
10 FOR I=1 TO 10
20 A=VAL(GETC("1234567890",I))
30 PRINT A:NEXT I
```

Конечное значение управляющей переменной — это первое число оператора DATA. Числа в аргументе оператора DATA взяты для примера, на практике они могут быть другими. Каждое нажатие клавиши EXE приводит к выводу на индикатор цифры, входящей в аргумент оператора GETC. Цифры выводятся в том порядке, в котором они записаны в аргументе оператора.

Ниже приведена программа нелинейного сглаживания при семи ординатах, при разработке которой использовано это свойство оператора GETC и формула

$$Y_i=(a_{i-3}Y_{i-3}+a_{i-2}Y_{i-2}+a_{i-1}Y_{i-1}+a_iY_i+a_{i+1}Y_{i+1}+ +a_{i+1}+a_{i+2}Y_{i+2}+ a_{i+3}Y_{i+3})/42.$$

Программа нелинейного сглаживания при семи ординатах

```
5 ! GETC-1
10 INPUT "N",N:FOR I=1 TO 7:
   PRINT"Y";I;: INPUT A(I):NEXT I
20 FOR I=0 TO 3:K=0
30 FOR J=1 TO 7
40 A$=GETC("30--00-0110--0-1110-0-01110-",
   7*I+J)
50 A=VAL(A$+GETC("78442428966474469
   224146 24264",7*I+J))
60 K=K+A(J)*A:NEXT J:PRINT
   CHR 23;I+1;"="; K/42:NEXT I
65 IF N=7 THEN 120
70 FOR I=8 TO N:FOR J=1 TO 6
80 A(J)=A(J+1):NEXT J
90 PRINT "Y";I:INPUT A(J)
100 K=-4*B+6*C+12*D+14*E+12*F+6*G-4*H
110 PRINT CHR 23;I-3;"=";K/42:NEXT I
120 FOR I=0 TO 2:K=0
130 FOR J=1 TO 7
140A$=GETC("0-0111-0--0110-00--03",
     7*I+J)
150 A=VAL(A$+GETC("142296447
     466982424488",7*I+J)
160 K=K+A(J)*A:NEXT J:PRINT CHR
     23;N-2+I; "=";K/42
170 NEXT I:END
```

Программа занимает в памяти 440 шагов.

На запрос N? — ввести число ординат, а на — YI? (I=1, 2,... 7, — величина I-й ординаты) ввести соответствующее число. После расчета выдается Ω — сглаженное значение I-й ординаты.

По сравнению с методом отдельного вычисления каждого значения y_i сэкономлено около 60 шагов.

А.Сербиенко (г.Киев)

ГРАФИЧЕСКИЙ РЕДАКТОР

Эта программа призвана облегчить работу по созданию изображений на экране МК-85. Она позволяет строить изображения, вводить и выводить изображения, представленные в виде кодов LETS позиций экрана, получать негативное изображение. Главным преимуществом программы является предоставляемая ею возможность сохранять изображение в памяти микрокомпьютера. Программа состоит из трех модулей: собственно редактора, графической памяти, программы создания негатива. Сообщения DEG, RAD и GRA указывают, какой именно модуль работает в данный момент времени. (DEG — работает редактор. RAD — работает программа обслуживания графической памяти. GRA — работает программа создания негатива.)

Запуск программы производится из файла редактора. В случае если в результате каких-либо действий пользователя графическая память (регистры A — L) оказалась занята цифровой информацией, при запуске может быть сообщение об ошибке. В этом случае программу следует запускать командой RUN 14. Функциональное назначение управляющих клавиш:

1,2,3,4,6,7,8,9 — движение "не отрывая карандаша".

5 — стереть точку.

. — выключить/включить графическую память. Всякая операция, кроме операций рисования, автоматически включает память.

AC — очистить графическую память и экран.

ANS — вывод изображения в кодах позиций экрана.

DEL — негатив изображения.

SPC — ввод изображения по кодам позиций экрана.

Нажатие любой другой клавиши вызовет на экран изображение, хранящееся в графической памяти.

Все команды воспринимаются программой только в режиме редактора. При необходимости координаты точки можно задать вручную, остановив программу командой STOP и присвоив их переменным X и Y.

Продуктивная работа программы возможна только в режиме повышенного быстродействия. Объем занимаемой памяти — 960 шагов.

Программа — Графический редактор

```
МОДУЛЬ РЕДАКТОРА.FILE 0 (#0)
1 T=1:MODE 4: CSR 0,:FOR M=0 TO 11:
  LETC A$(M):PRINT CHR 96;:NEXT M
2 N=ASCI KEY:IF N=0 THEN 2:IF N=7
  THEN 14:IF N=8 THEN #1
3 IF N=9 THEN 15:N=N-48:FOR M=1 TO 9:
  IF N=M THEN M+3:NEXT M:GOTO 16
4 IF X<=0 THEN 2:IF Y<=0 THEN 2:
  X=X-1: Y=Y-1:GOTO 13
5 IF Y<=0 THEN 2:Y=Y-1:GOTO 13
6 IF Y<=0 THEN 2:IF X>=59 THEN 2:
  X=X+1: Y=Y-1:GOTO 13
7 IF X<=0 THEN 2:X=X-1:GOTO 13
8 DRAWC X,Y:Z=0:IF T=1 THEN #2:
  GOTO 2
9 IF X>=59 THEN 2:X=X+1:GOTO 13
10 IF X<=0 THEN 2:IF Y>=6 THEN 2:
   X=X-1: Y=Y+1:GOTO 13
11 IF Y>=6 THEN 2:Y=Y+1:GOTO 13
12 IF X>=59 THEN 2:IF Y>=6 THEN 2:
   X=X+1: Y=Y+1
13 DRAW X,Y:Z=1:IF T=1 THEN #2:
   GOTO 2
14 FOR M=0 TO 11:A$(M)="(0)0000":
   NEXT M:  X=0:Y=0:GOTO 1
15 FOR M=0 TO 11:CSR 0,:
   PRINT M;":";A$(M):NEXT M:GOTO 1
16 IF N<>-16;IF N <>-2 THEN 1:T=-T:
   GOTO 2
17 FOR M=0 TO 11:CSR 0,:PRINT M;":";:
   INPUT A$(M):NEXT M:GOTO 1
```

```
МОДУЛЬ НЕГАТИВА.FILE P1(#1)
1 MODE 6:FOR M=0 TO 11:$=A$(M):
  FOR N=1 TO 7
2 P$=GETC(A$(M),N):O=ASCI P$
3 IF OO=O-48:GOTO 5
4 O=O-55
5 O=31-O
6 IF O;O=O+48:GOTO 8
7 O=O+55
8 P$=CHR O
9 $=MID(1,N-1)+P$+MID(N+1,7-N)
10 NEXT N:A$(M)=$:NEXT M:GOTO #0

МОДУЛЬ ГРАФИЧЕСКОЙ ПАМЯТИ.
FILE P2(#2)
1 MODE 5:$=GETC(A$(X/5),7-Y):
  N=ASCI $
2 IF N<=57 THEN 3:N=N-55:GOTO 4
3 N=N-48
4 FOR M=0 TO 4:S(4-M)=N-2*INT(N/2):
  N=INT(N/2):NEXT M
5 N=5*FRAC(X/5):S(N)=Z
7N=0:FOR M=0 TO 4:N=N+S(M)*2^(4-M):
  NEXT M
8 IF N<10 THEN 9:R$=CHR(N+55):
  GOTO 10
9 R$=CHR(N+48)
10 $=A$(X/5):IF Y<>6 THEN 11:O$="":
   P$=MID(2,6):GOTO 13
11 IF Y<>0 THEN 12:O$=MID(1,6):P$="":
   GOTO 13
12 O$=MID(1,6-Y):P$=MID(8-Y,Y)
13 A$(X/5)=O$+R$+P$:GOTO #0
```

В.Пономаренко (г.Москва)

ТЕЗЕЙ

Эта игра имитирует блуждания Тезея по трехмерному лабиринту. Кроме знаменитого клубка, Ариадна отдала ему свою губную помаду. Герой, приспособившись к слабому освещению, руководствуется подсказкой игрока, ожидая ее около 10 секунд (строка 61), но если подсказка запоздала, то нетерпеливый Тезей выбирает направление случайным образом, бросая игральный тетраэдр. Лабиринт состоит из 105 комнат с четырьмя дверьми в каждой, соединенных извилистыми переходами со спусками и

подъемами. Таким образом, план лабиринта не может быть представлен плоским графом. Всего, кроме входа и выхода, есть еще 418 дверей и 208 извилистых переходов. Все это создает МК-85 в режиме быстродействия за 20 секунд.

Тезею нужно найти ключ от выхода и выход. Еще одним ключом (запасным) он награждается в комнате № 105, если дойдет до нее.

Тезей придерживается следующей методики. Попав в комнату, в которой он еще не был, он присваивает ей порядковый номер и пишет его губной помадой на дверях, через которые вошел. А если он попадает в комнату, в которой уже был, то увидит этот номер на той двери, на которой он был написан. Местоположение этой двери укажет стрелка на индикаторе.

Управление осуществляется цифровыми клавишами: 1 — налево, 2 — направо, 3 — вперед, 0 — назад. Направление движения дублируется на индикаторе стрелкой справа от номера комнаты. О том, что выход найден, говорит появляющаяся на короткое время надпись: "ВЫХОД": Тезей выходит из лабиринта, если у него есть ключ.

При выходе Тезея из лабиринта программа останавливается. Нажав клавишу EXE, можно снова впустить Тезея в этот же лабиринт и попытаться найти дорогу к выходу за 7 ходов. Если не с первого, то с N-го раза это должно удастся. Если же программа запускается командой RUN, то будет построен новый лабиринт, и игра начнется сначала.

Игра складывается по-разному. Можно найти выход, не пройдя и 30 комнат, а потом долго искать ключ, а бывает наоборот. Дважды ключеносный Тезей долго ищет выход.

Программа ТЕЗЕЙ

```
1 DEFM 22:VAC:$="BDJCHNMFLGEKAIO":LETC"VV44767"
2 FOR A=1 TO 15:$=$+CHR A:NEXT A
3 FOR A=0 TO 6:FOR B=0 TO 4:D=A*5+B:N$(D)="":  FOR C=1 TO 3
4 E=ASCI GETC($,B*3+C)-49:N$(D)=N$(D)+GETC($,E):  NEXT C:NEXT B
5 FOR B=0 TO 4:$=MID(1,15+B*3)+N$(A*5+B):NEXT B: NEXT A
6 FOR A=0 TO 34:$="":FOR B=1 TO 3
7 $=$+CHR(ASCI GETC(N$(A),B)+INT(RAN#*6)*16): NEXT B
8 N$(A)=$+"000":NEXT A:A=1
11D=ASCIGETC(N$(I),A):E=INT(D/16):F=D-E*16
12G=INT(E/3):C=E-G*3:IF G=0;$="12312":GOTO 14
13 $="13213"
14 B$="0"+MID(C+1,3)
15 FOR C=1 TO 4:IF VAL GETC(B$,C)=L;M=C
16 NEXT C
21IFLEN(N$(I))<>7;N$(I)=N$(I)+"0":K=K+1
31D=ASCI GETC(N$(I),A+3)-48:IF D<>0 THEN 41
32J=J+1:D=J:$=N$(I);E=ASCI GETC($,7)+5^(A-1)*M
33 N$(I)=MID(1,A+2)+CHR (J+48)+MID(A+4,3-A)+CHR E: L=0:GOTO 51
41 E=ASCI GETC(N$(I),7)-48:FOR C=1 TO A:G=INT(E/5); L=E-G*5
42 E=G:NEXT C:L=L-M:IF L<0;L=L+4
51 $="↑←↑→":CSR0,:PRINT GETC($,L+1);D;
52 IF K=35;PRINT CHR 96;:IF J=105;PRINT CHR 96;
61 D=ASCI KEY:C=C+1:IF C>222;D=INT(RAN#*4+48)
62 IF D=0 THEN 61
63 D=D-48:IF D<0 THEN 61
64 IF D>3 THEN 61
71 H=H+1:PRINT GETC($,D+1);:M=M+D:IF M>4;M=M-4
72 L=VAL GETC(B$,M)
81 IF I+A+L=1;PRINT"ВХОД";:GOTO 11
82 IF I<>30 THEN 86
83 IF A+L<>A*L THEN 86
84 PRINT"ВЫХОД";:IF K<>35 THEN 11
85 PRINT H
86 L=L+2:IF L>3;L=L-4
91 M=0:D=-1^INT(L/2):IF FRAC(L/2)=0 THEN 95
92 A=A+D:IF A=0;A=3
```

```
93 IF A=4;A=1
94 GOTO 11
95 I=I+D:IF I=35;I=0
96 IF I<0;I=34
97 FOR C=1 TO 3:G=ASCI GETC(N$(I),C): IF G-INT(G/16)*16=F;M=C
98 NEXT C:IF M=0 THEN 95
99 A=M:GOTO 11
```

Любознательным читателям сообщаю, что алгоритм построения лабиринта связан с задачей Киркмана о 15 школьницах, опубликованной в книге М.Гарднера "Математические досуги" (М.: Мир, 1972, с. 12).

Вычислить вероятность прохождения лабиринта за семь ходов я не смог, но ее нижняя оценка не менее 1/729.

В МК-85м появляется возможность включить в игру активного Минотавра, охотящегося за Тезеем и управляемого по программе, либо другим игроком. Победителем тут будет напавший врасплох.

В.Вулис (Москва)

НА ВСЕ СЛУЧАИ ЖИЗНИ

За несколько лет эксплуатации МК-54 и МК-85 я обнаружил, что чаще всего приходится пользоваться двумя программами — поиском корней уравнения и поиском экстремума функции одной переменной. Приведенная ниже программа "Анализатор", по существу, является переводом на язык МК-85 моей программы для МК-54, опубликованной в журнале "Наука и жизнь" (№ 4/90, с. 96). Программа занимает в памяти 212 байт и, кроме того, необходимо место, начиная со строки 13, чтобы записать в программу анализируемую функцию.

Программа — "Анализатор

```
1 PRINT "13:Y(X),[ab]"
3 INPUT "a",X:GOSUB 13:PRINT Y:
INPUT " ↑>0, ↓=0",Z:IF Z>0 THEN 3
4 INPUT " ↓:-1,0, ↑:1",Z,"b",R,"e",S:
  R=R-X:Q=2:IF Z<>0;Q=1.618
5 U=Y
6 T=X
7 IF ABS R<S;PRINT"x=";T:GOTO 3
8 R=R/Q:X=R+T:GOSUB 13:IF Z=0
  THEN 11
9 IF (U-Y)*Z<0 THEN 5
10 R=-R:GOTO 7
11 IF U*Y>=0 THEN 6
12 GOTO 7
```

Строки имеют следующие назначения:

1 — напоминание, что в программе, начиная со строки 13, должна быть записана анализируемая функция.

2 — резервная, на случай, если анализируемая функция имеет параметры, ввод которых можно запрограммировать в этой строке.

3 — ввод значения "a" и вычисление функции в этой точке (опорное значение). Меню 0 — перейти к следующему этапу, любая другая цифра — вернуться к расчету другого опорного значения.

4 — меню: 0 — искать корень, положительное число — искать максимум, отрицательное — минимум. Ввод второго конца отрезка ("в") поиска и допустимой погрешности решения.

5 — присвоение опорного значения функции.

6 — присвоение нового значения аргументу.

7 — оценка достигнутой точности и вывод решения.

8 — уменьшение интервала, вычисление пробного значения аргумента, вычисление значения функции в пробной точке и отсылка его на сравнение с опорным.

9 — сравнение модулей (поиск экстремума).

10 — смена направления поиска при неудачной попытке.

11 — сравнение знаков (поиск корня...).

Следующая программа позволяет при отсутствии в памяти других программ решать методом Гаусса систему из N линейных уравнений. При использовании только 26 постоянно открытых регистров можно решать систему из пяти уравнений. Для шести уравнений нужно открыть еще семь регистров, для 7 — 15, для 8 — 24 и т.д.

Программа. Решение системы из N линейных уравнений методом Гаусса (1≤N≤15)

```
1 INPUT "[n]",A:B=A*(A+3)/4-10:
  DEFM B+ABS B:FOR B=0 TO A-1
2 FOR C=0 TO A:F(C)=0:NEXT C:
  FOR C=O TO A:CSR C:INPUT D
3 D=D+F(C):IF C>B THEN 7
4 IF C<B;FOR E=C+1 TO A:
  F(E)-D*F(E+(C+1)*(A-C/2)):
  NEXT E:NEXT C
5 IF D=0;PRINT CHR 3:GOTO 2
6 E=D:NEXT C
7 F(C+(B+1)*(A-B/2))=D/E:NEXT C:
  NEXT B:FOR B=0 TO A-1:FOR C=1 TO A-B
8D=A+C*(A-C/2+.5):F(D)=F(D)-F*F(D-B):
  NEXT C:F=F(D):E(C)=F:NEXT B
9 FOR C=1 TO A:PRINT C;"x=";F(C):
  NEXT C:END
```

"Линейная" система адресации регистров в МК-85 делает очень сложным формирование адресов элементов матрицы, именно на это была потрачена львиная доля времени при разработке программы.

Строки имеют следующее назначение:

1 — выделение памяти.

2 — очистка промежуточной памяти и ввод коэффициентов B-го уравнения. При вводе коэффициентов выполняется "прямой ход Гаусса".

4 — коэффициенты, стоящие на диагонали, приводятся к 1.

7 — занесение в память остальных коэффициентов.

Если диагональный элемент окажется нулем, то на индикатор выводится черный треугольник — символ плохой треугольной матрицы и необходимо ввести другое уравнение из еще не введенных (поменять порядок строк).

9 — вывод решения. Повторыный вывод решения может быть осуществлен подачей команды RUN 9.

Рамки статьи не позволили привести математическую основу метода Гаусса, но те, кто с ним незнаком, могут прочесть о нем в справочниках и учебниках по высшей математике, — например "Метод Гаусса исключения неизвестных" (А.А.Рывкин, А.З.Рывкин, Л.С.Хренов — Справочник по математике. — М.: Высшая школа, 1975. — С. 97 — 102).

ПРИМЕЧАНИЕ: TUTTI FRUTTI — в переводе с итальянского ВСЯКАЯ ВСЯЧИНА.

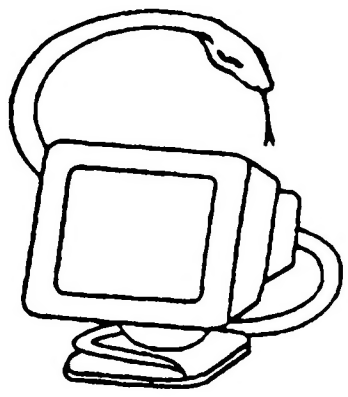
Уважаемые читатели, приславшие свои программы в редакцию. К сожалению, ограниченный объем выпуска не позволил довести до наших подписчиков все статьи, получившие одобрение. Автор и редактор выпуска намерены подготовить к печати книгу объемом 6 — 8 авторских листов. Следите за нашей рекламой.

СП
Диалог

Научный центр СП "Диалог" при ВЦ РАН

предлагает программные продукты,
разработанные сотрудниками СП "Диалог",
ВЦ Российской академии наук и других организаций

* Перечисленные ниже программные продукты могут быть приобретены за наличный расчет (при этом предоставляется скидка 20%), по предоплате или по договору.
* Для приобретения необходимого продукта по предоплате переведите требуемую сумму с указанием предмета платежа (например, "Оплата за программу AIDSTEST") на наш счет.
* Перешлите нам по почте, факсу или передайте лично копию платежного поручения с исполнением банка.
* Банковские реквизиты: **ПОЛУЧАТЕЛЬ** - НЦ СП "Диалог" при ВЦ РАН, текущий счет 345057; **БАНК ПОЛУЧАТЕЛЯ** - коммерческий банк "ДиалогБанк" в РКЦ ГУ ЦБР г.Москвы; **корреспондентский счет** 161625; **МФО** 201791; **код** 83.
* Почтовый адрес: 117967 Москва, ГСП-1, ул. Вавилова 40.
* Телефоны: (095) 137 0150 - директор, бухгалтер (и факс); 135 6253 - программные продукты и аппаратура; 137 0184 - антивирусные программы.
* Проезд: от станции метро "Ленинский проспект" любым трамваем или автобусом 115 до остановки "Улица Губкина" (4-я в направлении от центра города).
* BBS: (095) 135 6253 (с 20:00 до 10:00), FIDO 2:5020/6.13.



Предлагаем антивирусные программы
Д.Н. Лозинского и Д.Ю. Мостового

AIDSTEST и ADINF

Антивирус AIDSTEST Д. Н. Лозинского - одна из лучших отечественных программ для обнаружения и уничтожения компьютерных вирусов в операционной системе MS-DOS. В отличие от зарубежных антивирусных программ она ориентирована на вирусы, распространенные в нашей стране, что важно в связи с происходящим в России "вирусным взрывом". Последняя версия от 20 января 1992 г. распознает и уничтожает 225 различных вируса.. Среди них более половины созданы отечественными "технокрысами."

Антивирус ADINF Д .Ю. Мостового - одна из лучших отечественных программ-ревизоров для операционной системы MS-DOS. Принцип ее работы основан на том, что в специальных таблицах она сохраняет образы master-boot и boot-секторов диска, список номеров сбойных кластеров, схему дерева каталогов и информацию о всех контролируемых файлах. В настоящее время ADINF - единственный антивирус, который при загрузке

DOS с винчестера (а не с заклеенной дискеты, как того требуют другие программы) может обнаружить ЛЮБЫЕ типы вирусов. Кроме борьбы с вирусами он позволяет также следить за целостностью и сохранностью информации на винчестере и за всеми происходящими на диске изменениями, что особенно полезно при работе на широко распространенных в России "персональных ЭВМ коллективного пользования".

Для личного пользования индивидуальные покупатели за 25 руб. могут получить на свои дискеты копию текущей версии программ AIDSTEST или ADINF в Центре СП "Диалог" на ВДНХ в павильоне "Химия" ежедневно с 10 до 17 час. Телефоны : 181 9683, 181 9624. За 140 руб. можно купить программу на дискете и с документацией в специальной упаковке.

В Центре при ВЦ РАН организации могут купить текущие версии программ или оформить годовую подписку, что дает право получать на свои дискеты копии новых версий программ в течение двенадцати месяцев и использовать их на всех компьютерах в рамках организации. Дополнительно к этому подписка может включать пересылку версий программ на дискетах по почте. По отдельному договору может быть организовано приоритетное обслуживание - первоочередное изготовление в рамках очередной версии программы AIDSTEST средств для обнаружения и уничтожения новых (найденных Заказчиком) вирусов, консультации специалистов по анализу сложных ситуаций и т.п.

Варианты поставки для организаций	*Цена (руб.)*
AIDSTEST	
Копирование текущей версии на дискету Заказчика	*320*
Подписка - копирование на дискеты Заказчика	*1280*
Пересылка почтой на дискетах Поставщика (10 поставок, 1 раз в месяц, кроме июля -августа)	*1920*
Пересылка почтой на дискетах Поставщика (20 поставок, 2 раза в месяц, кроме июля--августа)	*2560*
ADINF	
Копирование текущей версии на дискету Заказчика	*640*
Пересылка почтой на дискете Поставщика	*704*
Подписка - копирование на дискеты Заказчика	*1024*
Пересылка почтой на дискетах Поставщика (4 поставки , раз в квартал)	*1280*

Цены действительны по состоянию на 20 января 1992г.

П 27 Персональный компьютер «Электроника МК 85». — М.: Знание, 1992. — 48 с. — (Новое в жизни, науке, технике. Сер. «Вычислительная техника и ее применение»; № 2).
ISBN 5-07-002247-4

Предлагаем читателям познакомиться с микро-ЭВМ "Электроника МК 85" и с БЕЙСИКом, закованным в его недрах. Представлен широкий подбор программ для МК 85 различного назначения.
Материал рассчитан на широкий круг читателей.

2404000000 **ББК 32.973**

ТЕМА СЛЕДУЮЩЕГО НОМЕРА:

МАШИННЫЕ КОДЫ

Научно-популярное издание

ПЕРСОНАЛЬНЫЙ КОМПЬЮТЕР «ЭЛЕКТРОНИКА МК 85»

Редактор *Б.М.Васильев*
Мл. редактор *Н.А.Васильева*
Художник *В.Н.Конюхов*
Худож.редактор *И.А.Емельянова*
Техн.редактор *Т.В.Луговская*
Корректор *В.И.Гуляева*

ИБ № 12186

Подписано к печати 30.01.92. Формат бумаги $70\times100^{1}/_{16}$. Бумага офсетная № 1. Печать офсетная. Усл. печ. л. 3,90. Усл. кр.-отт. 8,45. Уч.-изд. л. 4,29. Тираж 47400 экз. Заказ 2936. Издательство «Знание». 101835, ГСП, Москва, Центр, проезд Серова, д. 4. Индекс заказа 924702. Отпечатано с оригинал-макета издательства «Знание» на ордена Трудового Красного Знамени Тверском полиграфическом комбинате Министерства печати и массовой информации РСФСР. 170024, г. Тверь, пр. Ленина, 5.

Индекс 70195

Издательство «Знание»

Подписная научно-популярная серия

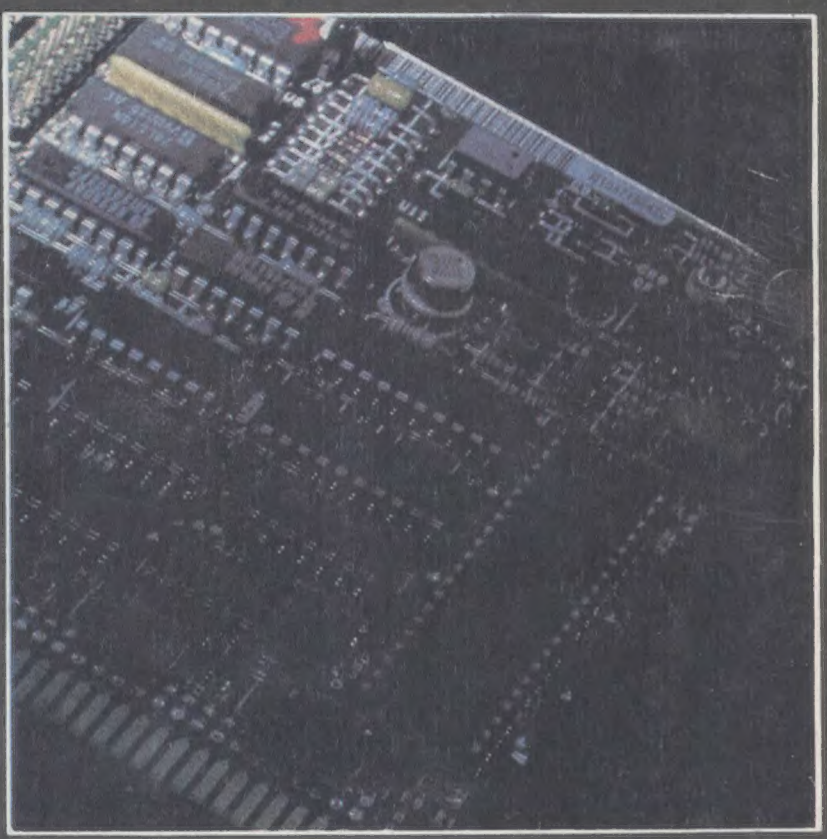
ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА И ЕЕ ПРИМЕНЕНИЕ

Будущее кроется в проектировании таких компьютеров, которых никто не примет за вычислительные машины

Адам Осборн, президент фирмы

Столько знаем, сколько в памяти удерживаем

Ф.И. Буслаев



Наш адрес: 101835, Москва, Центр, проезд Серова, 4